Yongjin Zhang, Lihong Feng, Suzhou Li and Peter Benner

Accelerating PDE constrained optimization by the reduced basis method: application to batch chromatography

MPIMD/14-09

May 26, 2014

Abstract

In this work, we show that the reduced basis method accelerates a PDE constrained optimization problem, where a nonlinear discretized system with a large number of degrees of freedom must be repeatedly solved during optimization. Such an optimization problem arises, for example, from batch chromatography. Instead of solving the full system of equations, a reduced model with a small number of equations is derived by the reduced basis method, such that only the small reduced system is solved at each step of the optimization process. An adaptive technique for selecting the snapshots is proposed, so that the complexity and runtime for generating the reduced basis are largely reduced. An output-oriented error bound is derived in the vector space whereby the construction of the reduced model is managed automatically. An early-stop criterion is proposed to circumvent the stagnation of the error and to make the construction of the reduced model more efficient. Numerical examples show that the adaptive technique is very efficient in reducing the offline time. The optimization based on the reduced model is successful in terms of the accuracy and the runtime for getting the optimal solution.

Keywords: reduced basis method, empirical interpolation, adaptive snapshot selection, optimization, batch chromatography

1 Introduction

In the last decade, the optimization with constraints given by partial differential equations (PDE constrained optimization, for short), has emerged as a challenging research area. It has increasingly arisen in various engineering contexts, such as optimal design, control and parameter estimation. Over the past years, besides the increasing progress of the computing hardware, a large number of attempts have been devoted to the development of efficient algorithms and strategies for solving such optimization problems, see for example [6, 7, 8, 26] and references therein.

Model order reduction (MOR) is a powerful technique for constructing a low-cost approximation of large-scale systems resulting from the discretization of PDEs. The low-cost approximation, often called reduced order model (ROM), on the one hand, should have the same structure as the original large-scale system, but with a much smaller number of degrees of freedom (DOFs); on the other hand, it must have acceptable accuracy for the input-output representation of the original system. Due to the small size and negligible error, the derived ROM is used as a surrogate model of the large-scale system in various disciplines, such as optimization and control, fluid dynamics, structural dynamics, circuit design, and so on. In particular, for optimization problems with nonlinear PDE constraints, proper orthogonal decomposition (POD) is often used to derive a ROM, which has been applied to accelerate optimization problems [13, 14]. However, a ROM from POD is reliable only in the neighborhood of the input parameter setting at which the ROM is constructed. There is no guarantee for the accuracy of the ROM at a different parameter setting. To circumvent the problem, a trust-region technique was suggested to manage the POD-based ROM in [13]. Here, the ROM is updated according to the quality of the approximation. However, the repeated construction of the ROM reduces the significance of the reduction in computational resources obtained by MOR. In contrast, the technique of parametric model order reduction (PMOR) enables the generation of a parametric ROM with acceptable accuracy over the feasible parameter domain, such that a single ROM is sufficient for the optimization process. Among the various PMOR methods [1, 3, 5, 10, 15, 16], few of them are applicable for nonlinear problems with parameters. The reduced basis method (RBM), however, has been developed for nonlinear parametric systems [2, 11, 18, 34]. Moreover, endowed with a posteriori error estimation, the parametric ROM can be generated automatically.

The RBM has been proved to be powerful tools for rapid and reliable evaluation of the parameterized PDEs [2, 11, 18, 34]. The reduced basis (RB), used to construct the ROM, is computed from snapshots (the solutions of the PDEs at certain selected samples of the parameters and/or chosen time steps) through a greedy algorithm. When applied to optimization, the original system resulting from the discretization of PDEs is first replaced by a ROM generated by the RBM, then the related quantities can be evaluated rapidly by solving the cheap ROM rather than the original expensive one. So far, research on the application of RBMs to PDE constrained optimization is very limited. In [33], the authors mainly focused on RBMs for affinely parameterized linear problems. Shape optimization employing RBMs for viscous flow in hemodynamics was addressed in [29], where the empirical interpolation method (EIM) [2] was exploited to treat the nonaffinity in the linear parameterized system. Applications to multiscale problems can be found in the recent work [32]. However, all these applications focus on finite element (FE) based RBMs for linear time-independent PDEs.

In this paper, we consider an optimization problem with PDE constraints, where the PDEs are nonlinear, time-dependent and have non-affine parameter dependency. Such problems arise, for example, from batch chromatography in chemical engineering. To capture the dynamics precisely, a large number of DOFs must be employed, which results in a large-scale system. Solving such a complex system during optimization is time-consuming. Constructing a reduced model for a parameterized nonlinear, time-dependent, nonaffine system poses additional challenges for all kinds of MOR methods, granting no exemption to RBMs. Furthermore, a careful choice of the discretization scheme should be taken for nonlinear problems, especially for convection-dominated problems. The finite volume (FV) discretization is used to construct the full order model (FOM), by which the conservation property of the system is well preserved. The FV-based RBM was first introduced for linear evolution equations in [24], and is extended to nonlinear problems afterwards [11, 25], where the nonlinear operator resulting from the discretization will be treated with empirical operator interpolation for an efficient offline-online computation of the ROM.

With no doubt, an efficient, rigorous and sharp a posteriori error estimation is crucial for RBMs because it enables automatic generation of the RB, and in turn a reliable ROM with a desired accuracy, with the help of a greedy algorithm. Rapid and reliable evaluation of the input-output relationships for the associated PDEs is very important for efficiently solving the optimization problem, where an output response rather than the field variable (the solution to the PDEs) is of interest. When a ROM is employed for such an evaluation, the error of the output of interest rather than that of the field variable, should be estimated and used for the generation of the ROM. We propose to use the output error for the generation of the RB. There are some results on the output error bound for FE-based RBMs for elliptic or parabolic problems [35, 36]. However, there is no study on the output error bound for FV-based RBMs for nonlinear evolution equations so far. In this work, we present a residual-based error estimation for the output of the ROM derived by a FV-based RBM to obtain a goal-oriented ROM.

With the help of an error estimate, the construction of the ROM can be managed automatically. In some cases, however, the error bound may not work as well as one expects. For example, in the process of the basis being extended, the error bound decreases slowly or even stagnates after some steps but the true error is very small already. As a result, the basis extension is not stopped because the error bound does not go below the prespecified tolerance. This means that the basis will be unnecessarily extended if there is no reasonable remedy. Certainly, simply using the true error as the indicator is not a wise choice because it is typically time-consuming to compute the true error for all sample points in the training set. To make full use of the available error estimate, we propose an early-stop criterion for the basis extension by checking the true error at the parameter selected by the greedy algorithm according to the output error bound. In this way, the basis extension can be stopped in time and the size of the resulting ROM can be kept reasonably small.

Additionally, the efficiency of the RBM is ensured by the strategy of offline-online decomposition. During the offline stage, all full-dimension dependent and parameter-independent terms can be precomputed and a parameterized reduced model is obtained a priori; during the process of optimization, a reliable output response can be obtained rapidly by the online simulation based on the ROM at the parameter determined by the optimization procedure. In this way, the ROM-based optimization can be solved more efficiently compared to the FOM-based one. Note that the offline time is usually not taken into consideration, although the offline computation is typically time-consuming, especially for time-dependent PDEs.

To reduce the cost and complexity of the offline stage, we propose a technique of adaptive snapshot selection (ASS) for the generation of the RB. For time-dependent problems, if the dynamics (rather than the solution at the final time) is of interest, the solution at the time instances in the evolution process should be collected as snapshots. However, the trajectory for a given parameter might contain a large number of time steps, e.g. in the simulation of batch chromatography. In such a case, if the solutions at all time steps are taken as snapshots, the subsequent computation will be very expensive because the number of snapshots is too large; if one just trivially selects part of the solutions, i.e., solutions at parts of the time instances (e.g. every two or several time steps), the final RB approximation might be of low accuracy because important information may have been lost due to such a naive snapshot selection. We propose to select the snapshot adaptively according to the variation of the solution in the evolution process. The idea is to make full use of the behavior of the trajectory and discard the redundant (linearly dependent) information adaptively. It enables the generation of the RB with a small number of snapshots but including only "useful" information. In addition, it is easily combined with other algorithms for the generation of the RB, e.g. the POD-Greedy algorithm [24].

This paper is organized as follows. We state the underlying PDE-constrained optimization problem in detail in Section 2. Reviews of the RBM and EIM are given in Section 3 and Section 4, respectively. The adaptive technique of snapshot selection and its implementation are addressed in detail in Section 5. Section 6 shows the RB scheme for the batch chromatographic model, including the derivation of the FOM based on the FV discretization, the generation of the ROM, and the strategy of the offline-online decomposition as well. In Section 7, an output-oriented error bound is derived in the vector space for evolution equations for the RBM based on FV-discretization. An early-stop criterion is proposed to make the construction of the ROM more efficient. Numerical examples including optimization based on the ROM are carried out in Section 8. Conclusions are drawn in Section 9.

2 Problem statement

In this work, we consider the following PDE constrained optimization problem:

$$\begin{aligned} &\min_{\mu\in\mathcal{P}} \left\{\mathcal{J}\left(u(t,x;\mu);\mu\right)\right\}, \\ s.t. \quad &\Psi\left(u(t,x;\mu);\mu\right) \le 0, \\ &\Phi\left(u(t,x;\mu);\mu\right) = 0, \end{aligned} \tag{1}$$

where $\mathcal{J}$ is the objective function, Ψ defines the inequality constraints. The field variable $u(t,x;\mu)$ is the solution to the underlying parametrized partial differential equations $\Phi\left(u(t,x;\mu)\right)=0$, $\mu \in \mathcal{P}$. Such an optimization problem arises in many applications, such as aerodynamics, fluid dynamics and chemical process. In practical computation, the PDEs are usually discretized such that the optimization problem in (1) is replaced by an optimization problem in finite dimensions:

$$\begin{aligned} &\min_{\mu\in\mathcal{P}} \left\{\tilde{\mathcal{J}}(u^{\mathcal{N}}(t,\mu);\mu)\right\}, \\ s.t.\ &\tilde{\Psi}\left(u^{\mathcal{N}}(t,\mu);\mu\right) \le 0, \\ &\tilde{\Phi}\left(u^{\mathcal{N}}(t,\mu);\mu\right) = 0, \end{aligned} \tag{2}$$

where $u^{\mathcal{N}} := u^{\mathcal{N}}(t,\mu) \in \mathbb{R}^{\mathcal{N}}$ is the solution to the discretized system of equations $\tilde{\Phi}\left(u^{\mathcal{N}}(t,\mu);\mu\right)=0$, and $\tilde{\mathcal{J}}, \tilde{\Psi}$ and $\tilde{\Phi}$ are the operators in the finite dimensional vector space corresponding to $\mathcal{J}, \Psi$ and Φ, respectively. The discretized equations are often of very large scale and complex. At each iteration of the optimization process, such a large-scale complex system of equations must be solved at least once. As a result, the whole optimization process will be time-consuming. To accelerate the underlying optimization, a surrogate ROM can be employed to replace the original large-scale discretized system for a rapid evaluation of the vector of unknowns $u^{\mathcal{N}}$.

To further motivate and illustrate our methods, we consider a particular example: optimal operation for batch chromatography. Batch chromatography, as a crucial separation and purification tool, is widely employed in food, fine chemical and pharmaceutical industries. The principle of batch elution chromatography for binary separation is shown schematically in Figure 1. During the injection period t_{in}, a mixture consisting of a and b is injected at the inlet of the column packed with a suitable stationary phase. With the help of the mobile phase, the feed mixture flows through the column. Since the solutes to be separated exhibit different adsorption affinities to the stationary phase, they move at different velocities in the column, and thus separate from each other when exiting the column. At the column outlet, component a is collected between cutting points t_3 and t_4, and component b is collected between t_1 and t_2. Here the positions of t_1 and t_4 are determined by a minimum concentration threshold that the detector can resolve, and the positions of t_2 and t_3 are determined by the purity specifications (Pu_{a} and Pu_{b}) imposed on the products. After a cycle period $t_{cyc} := t_4 - t_1$, the injection is repeated.

The dynamic behavior of the chromatographic process is described by an axially dispersed plug-flow model with limited mass-transfer rate characterized by a linear driving force approximation. The governing equations in the dimensionless form are formulated as follows,

$$\begin{cases} \dfrac{\partial c_z}{\partial t} + \dfrac{1-\epsilon}{\epsilon}\dfrac{\partial q_z}{\partial t} = -\dfrac{\partial c_z}{\partial x} + \dfrac{1}{Pe}\dfrac{\partial^2 c_z}{\partial x^2}, & 0 < x < 1, \\ \qquad\qquad\quad \dfrac{\partial q_z}{\partial t} = \dfrac{L}{Q/(\epsilon A_c)}\kappa_z\left(q_z^{\text{Eq}} - q_z\right), & 0 \le x \le 1, \end{cases} \tag{3}$$

where c_z, q_z are the concentrations of the component z ($z = \mathsf{a}, \mathsf{b}$) in the liquid and solid phase, respectively, Q the volumetric feed flow-rate, A_c the cross-sectional area of the column with the length L, ϵ the column porosity, κ_z the mass-transfer coefficient, and Pe the Péclet number. The adsorption equilibrium q_z^{Eq} is described by the isotherm equations of bi-Langmuir type,

$$q_z^{\mathrm{Eq}} = f_z(c_{\mathsf{a}}, c_{\mathsf{b}}) := \frac{H_{z1}c_z}{1 + K_{\mathsf{a}1}c_{\mathsf{a}}^{\mathrm{f}}c_{\mathsf{a}} + K_{\mathsf{b}1}c_{\mathsf{b}}^{\mathrm{f}}c_{\mathsf{b}}} + \frac{H_{z2}c_z}{1 + K_{\mathsf{a}2}c_{\mathsf{a}}^{\mathrm{f}}c_{\mathsf{a}} + K_{\mathsf{b}2}c_{\mathsf{b}}^{\mathrm{f}}c_{\mathsf{b}}}, \tag{4}$$

where c_z^{f} is the feed concentration of component z, H_{zj} and K_{zj} are the Henry constants and thermodynamic coefficients, respectively. The initial and boundary conditions are given as follows:

$$\begin{cases} c_z(0,x) = 0, \quad q_z(0,x) = 0, \quad 0 \le x \le 1, \\ \dfrac{\partial c_z}{\partial x}|_{x=0} = Pe\left(c_z(t,0) - \chi_{[0,t_{\mathrm{in}}]}(t)\right), \\ \dfrac{\partial c_z}{\partial x}|_{x=1} = 0, \end{cases} \tag{5}$$

where t_{in} is the injection period, and $\chi_{[0,t_{\mathrm{in}}]}$ is the characteristic function,

$$\chi_{[0,t_{\mathrm{in}}]}(t) = \begin{cases} 1, & \text{if } t \in [0, t_{\mathrm{in}}]; \\ 0, & \text{otherwise.} \end{cases}$$

More details about the mathematical modeling for batch chromatography can be found in [20].

Note that the feed flow rate Q and the injection period t_{in} are often considered as the operating variables, denoted as $\mu := (Q, t_{\mathrm{in}})$, which play the role of parameters in the PDEs (3)−(5). The system of PDEs is nonlinear, time-dependent and has non-affine parameter dependency. The nonlinearity of the system is reflected by (4). To capture the system dynamics precisely, a large number of DOFs must be introduced for the discretization of the PDEs.

The optimal operation of batch chromatography is of practical importance since it allows to exploit the full economic potential of the process and to reduce the separation

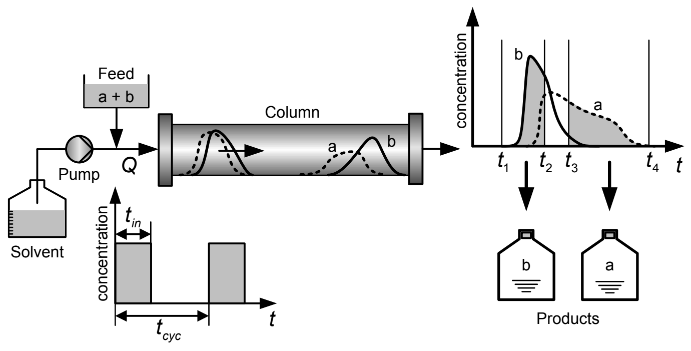

Figure 1: Sketch of a batch chromatographic process for the separation of a and b.

cost. Many efforts have been made for the optimization of batch chromatography over the past several decades. An extensive review of the early work can be found in [20] and references therein. An iterative optimization approach for batch chromatography was addressed in [17]. A hierarchical approach to optimal control for a hybrid batch chromatographic process was developed in [19]. Notably, all these studies are based on the finely discretized FOM. Such a model with a large number of DOFs is able to capture the dynamics of the process, and the accuracy of the optimal solution obtained from that can be guaranteed. However, the expensive FOM must be repeatedly solved in the optimization process, which makes the runtime for obtaining the optimal solution rather too long.

In this work, the RBM is employed to generate a surrogate ROM of the parameterized PDEs. The resulting ROM is used to get a rapid evaluation of the output response $y(u^{\mathcal{N}})$ for the discretized system $\tilde{\Phi}(u^{\mathcal{N}}(t,\mu);\mu) = 0$ in (2) in the optimization process. In the next section, we review the RBM and highlight some difficulties there.

3 Reduced basis methods

Reduced basis methods, first introduced in the late of 1970s for nonlinear structural analysis [31], have gained increasing popularity for parameterized PDEs in the last decade [18, 24, 34]. The basic assumption of RBMs is that the solution to parametrized PDEs, $u(\mu)$, depends smoothly on the parameter μ in the parameter domain $\mathcal{P}$, such that for any parameter $\mu \in \mathcal{P}$, the corresponding solution $u(\mu)$ can be well approximated by a properly precomputed basis, called reduced basis. In addition, the RBM is often endowed with a posteriori error estimation, which is used for the qualification of the resulting ROM.

Consider a parametrized evolution problem defined over the spatial domain $\Omega \subset \mathbb{R}^d$ and the parameter domain $\mathcal{P} \subset \mathbb{R}^p$,

$$\partial_t u(t,x;\mu) + \mathcal{L}[u(t,x;\mu)] = 0, \quad t \in [0,T], \quad x \in \Omega, \quad \mu \in \mathcal{P}, \tag{6}$$

where $\mathcal{L}[\cdot]$ is a spatial differential operator. Let $\mathcal{W}^{\mathcal{N}} \subset L^2(\Omega)$ be an $\mathcal{N}$-dimensional discrete space in which an approximate numerical solution to equation (6) is sought. Let $0 = t^0 < t^1 < \cdots < t^K = T$ be $K+1$ time instants in the time interval $[0,T]$. Given $\mu \in \mathcal{P}$ with suitable initial and boundary conditions, the numerical solution at the time $t = t^n$, $u^n(\mu)$, can be obtained by using suitable numerical methods, e.g. the finite volume method. Assume that $u^n(\mu) \in \mathcal{W}^{\mathcal{N}}$ satisfies the following form,

$$L_I(t^n)[u^{n+1}(\mu)] = L_E(t^n)[u^n(\mu)] + g(u^n(\mu),\mu), \tag{7}$$

where $L_I(t^n)[\cdot], L_E(t^n)[\cdot]$ are linear implicit and explicit operators respectively, and $g(\cdot)$ is a nonlinear μ-dependent operator. These operators are obtained from the discretization of the time derivative and spatial differential operator $\mathcal{L}$. For implicit scheme of FVMs, $L_I(t^n)$ can be nonlinear, see e.g. [11], but we only consider the linear case in this paper. By convention, $u^n(\mu)$ is considered as the "true" solution by assuming that the numerical solution is a faithful approximation of the exact (analytical) solution $u(t^n,x;\mu)$ at the time instance t^n.

The RBM aims to find a suitable low dimensional subspace

$$\mathcal{W}^N = \operatorname{span}\{V_1, \dots, V_N\} \subset \mathcal{W}^{\mathcal{N}},$$

and solve the resulting ROM to get the RB approximation $\hat{u}^n(\mu)$ to the "true" solution $u^n(\mu)$. In addition or alternatively to the field variable itself, the approximation of outputs of interest can also be obtained inexpensively by $\hat{y}(\mu) := y\left(\hat{u}(\mu)\right)$. More precisely, given a matrix $V := [V_1, \dots, V_N]$, whose columns span the reduced basis, the Galerkin projection is employed to generate the ROM as follows:

$$V^T L_I(t^n)[V a^{n+1}(\mu)] = V^T L_E(t^n)[V a^n(\mu)] + V^T g\left(V a^n(\mu)\right), \tag{8}$$

where $a^n(\mu) := \left(a_1^n(\mu), \dots, a_N^n(\mu)\right)^T \in \mathbb{R}^N$ is the vector of the weights in the approximation $\hat{u}^n(\mu) := V a^n(\mu) = \sum_{i=1}^N a_i^n(\mu) V_i$, and it is the vector of unknowns in the ROM. Thanks to the linearity of the operators L_I and L_E, the ROM (8) can be rewritten as

$$V^T L_I(t^n) V[a^{n+1}(\mu)] = V^T L_E(t^n) V[a^n(\mu)] + V^T g\left(V a^n(\mu)\right), \tag{9}$$

where $V^T L_I(t^n) V$ and $V^T L_E(t^n) V$ can be precomputed and stored for the construction of the ROM. However, the computation of the last term of (9), $V^T g\left(V a^n(\mu)\right)$, cannot be done analogously because of the nonlinearity of g. This will be tackled by using a technique of empirical interpolation, to be addressed in the next section.

How to generate the RB V is crucial, and is still an active field of study. A popular algorithm for the generation of the RB for time-dependent problems is the POD-Greedy algorithm [24], as is shown in Algorithm 1.

Algorithm 1 RB generation using POD-Greedy

```
Input:  $\mathcal{P}_{\text{train}}$, $\mu_0$, $tol_{\text{RB}} (< 1)$
Output: RB $V = [V_1, \dots, V_N]$
1: Initialization: $N = 0$, $V = [\,]$, 0 $\mu_{\max} = \mu_0$, $\eta_N(\mu_{\max}) = 1$
2: while $\eta_N(\mu_{\max}) > tol_{\text{RB}}$ do
3:     Compute the trajectory $S_{\max} := \{u^n(\mu_{\max})\}_{n=0}^K$.
4:     Enrich the RB, e.g. $V := [V, V_{N+1}]$, where $V_{N+1}$ is the first POD mode of the
       matrix $\bar{U} := [\bar{u}^0, \dots, \bar{u}^K]$ with $\bar{u}^n := u^n(\mu_{\max}) - \Pi_{\mathcal{W}^N}[u^n(\mu_{\max})], n = 0, \dots, K$.
       $\Pi_{\mathcal{W}^N}[u]$ is the projection of $u$ onto the current space $\mathcal{W}^N := \operatorname{span}\{V_1, \dots, V_N\}$.
5:     $N = N + 1$
6:     Find $\mu_{\max} := \arg\max_{\mu \in \mathcal{P}_{\text{train}}} \eta_N(\mu)$.
7: end while
```

Remark 3.1. In Algorithm 1, the error $\eta_N(\mu_{\max})$ is an indicator for the error of the ROM. It can be the true error or an error estimation. Since the true error requires the "true" solution $u^n(\mu)$ by solving the full large system, an error bound is usually used instead. This is explored in Section 7. The first POD mode refers to the first left singular vector which corresponds to the largest singular value of the matrix under consideration.

Remark 3.2. As is mentioned, an error bound is usually used as the indicator in Algorithm 1 since it is much cheaper to compute in comparison with the true error, but it may not always work well. For example, in the process of the basis being extended, the error bound decreases slowly or even stagnates after some steps. To circumvent this problem, we propose a remedy by checking the true error at the parameter determined by the greedy algorithm and get an early-stop for the extension of the RB. Details are given in Algorithm 5 in Section 7.3.

The theoretical analysis about the convergence of the POD-Greedy algorithm is given in the recent work [21]. However, for some problems, such as batch chromatography, the implementation of Step 4 in Algorithm 1 will be time-consuming because the number of time steps K needs to be very large due to the nature of the problem (integration until a certain steady state is reached). In this work, we propose to use an adaptive technique to reduce the cost of Step 4, which is discussed in Section 5.

4 Empirical interpolation

As mentioned above, if there are nonlinear and/or nonaffine operators in the full model, the computational complexity cannot be reduced by using projection, because the nonlinear and/or nonaffine part, e.g. $V^T g\left(V a^n(\mu)\right)$ in (8), requires the computation in the original full space. In such a case, EIM [2] or empirical operator interpolation [11] can be exploited to generate an efficient ROM. The empirical operator interpolation method is an extension of the EIM, and can be used to treat an operator which depends on the parameter, field variable and spatial variable as well. The idea of EIM, introduced in [2], is briefly presented as follows.

Given a nonaffine μ–dependent function $g(x,\mu)$ with sufficient regularity, $(x,\mu) \in \Omega \times \mathcal{P} \subset \mathbb{R}^d \times \mathbb{R}^p$, the idea of EIM is to approximate $g(x,\mu)$ by a linear combination of a precomputed μ-independent basis $W := [W_1, \dots, W_M]$, termed as collateral reduced basis (CRB), with corresponding μ-dependent coefficients $\sigma(\mu) := [\sigma_1(\mu), \dots, \sigma_M(\mu)]^T$, i.e.,

$$\hat{g}(x,\mu) = \sum_{i=1}^{M} W_i(x)\sigma_i(\mu).$$

Here the coefficients σ_i are parameter-dependent and determined by solving the linear system:

$$g(x_j,\mu) = \sum_{i=1}^{M} W_i(x_j)\sigma_i(\mu), \quad j = 1, \dots, M, \tag{10}$$

where $W_i(x_j)$ refers to the j-th entry of the vector W_i, and the analogous notation is also used for $\xi_m(x_m)$ in (11) in Algorithm 2. Note that the approximation $\hat{g}(x,\mu)$ interpolates the exact value $g(x,\mu)$ at the EI points $T_M := \{x_1, \dots, x_M\}$. The generation of the CRB and the EI points is illustrated in Algorithm 2.

Remark 4.1. Algorithm 2 is used for a fast evaluation of a nonaffine function of the coordinate x and the parameter μ by using interpolation. In [11, 25], the idea

Algorithm 2 Generation of CRB and EI points

```
Input: $L_{\text{train}}^{\text{crb}} := \{g(x,\mu) \mid \mu \in \mathcal{P}_{\text{train}}^{\text{crb}}\}, tol_{\text{CRB}}(< 1)$
Output: CRB $W = [W_1, \dots, W_M]$ and EI points $T_M = \{x_1, \dots, x_M\}$
1:  Initialization: $m = 1, \mathcal{W}_{EI}^0 := [\,], \|\xi_0\| = 1$
2:  while $\|\xi_{m-1}\| > tol_{\text{CRB}}$ do
3:    For each $g \in L_{\text{train}}^{\text{crb}}$, compute the "best" approximation $\hat{g} := \sum_{i=1}^{m-1} \sigma_i W_i$ in
      the current space $\mathcal{W}_{EI}^{m-1} := \text{span}\{W_1, \dots, W_{m-1}\}$, where $\sigma_i$ can be obtained by
      solving the linear system (10).
4:    Define $g_m := \arg \max_{g \in L_{\text{train}}^{\text{crb}}} \|g - \hat{g}\|$, and the error $\xi_m := g_m - \hat{g}_m$.
5:    if $\|\xi_m\| \le tol_{\text{CRB}}$ then
6:      Stop and set $M = m - 1$.
7:    else
8:      Determine the next EI point and basis:
          $$x_m := \arg \sup_{x \in \Omega} |\xi_m(x)|\,, \; W_m := \frac{\xi_m}{\xi_m(x_m)}. \qquad (11)$$
9:    end if
10:   $m := m + 1$
11: end while
```

was extended to the more general case of empirical operator interpolation, which is more applicable for an operator that depends on the field variable $u(t, x; \mu)$, e.g. $g(u(t, x; \mu), x; \mu)$. The evaluation of $g(x_j, \mu)$ in (10) is thus replaced by $g(u(t, x_j; \mu), x_j; \mu)$. In this paper, we use empirical operator interpolation, where the nonaffine operator appears as $g(u(t, x; \mu); \mu)$. The details are addressed in Section 6.2.

5 Adaptive snapshot selection

In this section, we propose a technique of adaptive snapshot selection we call ASS to reduce the offline cost. The basic idea of ASS is first presented in the ENUMATH2013 conference, and the following algorithms, e.g., Algorithm 3 and Algorithm 4, can be also found in [4]. In this paper, we address the ASS technique with more details, and enhanced numerical results are given in Section 8.

For the generation of the RB or CRB, a training set $\mathcal{P}_{\text{train}}$ or $\mathcal{P}_{\text{train}}^{\text{crb}}$ of parameters must be determined. On the one hand, the size of the training set is desired to be as large as possible, so that information of the parametric system can be collected as much as possible. On the other hand, the RB or CRB should be efficiently generated.

To reduce the cost for the generation of the RB, many efforts have been made in the last decade. These include, for example, the *hp* certified RB method [12], adaptive grid partition in parameter space [22, 23], and the greedy-based adaptive sampling approach for MOR by using model-constrained optimization [9]. In these papers, the authors intend to choose the sample points adaptively and get an "optimal" training set. The "optimal" training set means that the original manifold $\mathcal{M} := \{u(\mu) \mid \mu \in \mathcal{P}\}$ can be well represented by the submanifold $\hat{\mathcal{M}} := \{u(\mu) \mid \mu \in \mathcal{P}_{\text{train}}\}$ induced by the sample set $\mathcal{P}_{\text{train}}$ with its size as small as possible.

As aforementioned, for time-dependent problems, if the dynamics is of interest, the solution at the time instances should be collected as snapshots. In such a case, even for an "optimal" training set, the number of snapshots can be huge if the total number of time steps for a single parameter is large. Such problems may arise from, e.g. chemical engineering, fluid dynamics and aerodynamics etc.. A large number of snapshots means that it is time-consuming to generate the reduced basis because the POD mode in Step 4 in Algorithm 1 is hard to compute from the singular value decomposition of $\bar{U}$, due to the large size of the matrix $\bar{U}$. This is also true for the generation of the CRB if the operator to be approximated is time-dependent. As a straightforward way to avoid using the solutions at all the time instances as snapshots, one can simply pick out the solutions at certain time instances (e.g., every two or several time steps) as snapshots. However, the results might be of low accuracy, because some important information may have been lost during such a trivial snapshot selection.

For an "optimal" or a selected training set, we propose to select the snapshots adaptively according to the variation of the trajectory of the solution, $\{u^n(\mu)\}_{n=0}^{K}$. The idea is to discard the redundant ("close to" linearly dependent) information from the trajectory. In fact, the linear dependency of two non-zero vectors v_1 and v_2 can be reflected by the angle θ between them. More precisely, they are linearly dependent if and only if $|\cos(\theta)| = 1$ ($\theta = 0$ or π). In other words, the value $1 - |\cos(\theta)|$ is large if the correlation between the two vectors is weak. This implies that the quantity $1 - \frac{|\langle v_1, v_2\rangle|}{\|v_1\|\|v_2\|}$ ($\cos(\theta) = \frac{\langle v_1, v_2\rangle}{\|v_1\|\|v_2\|}$) is a good indicator for the linear dependency of v_1 and v_2.

Given a parameter μ and the initial vector $u^0(\mu)$, the numerical solution $u^n(\mu)$ ($n = 1, \ldots, K$) can be obtained, e.g. by using the evolution scheme (7). Define an indicator $Ind\,(u^n(\mu), u^m(\mu)) = 1 - \frac{|\langle u^n(\mu), u^m(\mu)\rangle|}{\|u^n(\mu)\|\|u^m(\mu)\|}$, which is used to measure the linear dependency of the two vectors. When $Ind\,(u^n(\mu), u^m(\mu))$ is large, the correlation between $u^n(\mu)$ and $u^m(\mu)$ is weak. Algorithm 3 shows the realization of the ASS, $u^n(\mu)$ is taken as a new snapshot only when $u^n(\mu)$ and $u^{n_j}(\mu)$ are "sufficiently" linearly independent, by checking whether $Ind\,(u^n(\mu), u^{n_j}(\mu))$ is large enough or not. Here, $u^{n_j}(\mu)$ is the last selected snapshot.

Remark 5.1. The inner product $\langle \cdot, \cdot \rangle : \mathcal{W}^{\mathcal{N}} \times \mathcal{W}^{\mathcal{N}} \to \mathbb{R}$ used above is properly defined according to the solution space $\mathcal{W}^{\mathcal{N}}$, and the norm $\|\cdot\|$ is induced by the inner product correspondingly. Therefore, the ASS technique is applicable to any snapshot based MOR method for time-dependent problems, and it is independent of the discretization method.

Remark 5.2. For the linear dependency, it is also possible to check the angle between the tested vector $u^n(\mu)$ and the subspace spanned by the selected snapshots S^A. More redundant information can be discarded but at higher cost. However, the data will be compressed further, e.g. by using the POD-Greedy algorithm, we simply choose the economical case shown in Algorithm 3. Note that the tolerance tol_{ASS} is prespecified and problem-dependent, and the value at $O(10^{-4})$ gives good results for the numerical

Algorithm 3 Adaptive snapshot selection (ASS)

```
Input:    Initial vector $u^0(\mu)$, $tol_{\text{ASS}}$
Output: Selected snapshot matrix $S^A = [u^{n_1}(\mu), u^{n_2}(\mu), \dots, u^{n_\ell}(\mu)]$
1: Initialization: $j = 1$, $n_j = 0$, $S^A = [u^{n_j}(\mu)]$
2: for $n = 1, \dots, K$ do
3:    Compute the vector $u^n(\mu)$.
4:    if $Ind(u^n(\mu), u^{n_j}(\mu)) > tol_{\text{ASS}}$ then
5:       $j = j + 1$
6:       $n_j = n$
7:       $S^A = [S^A, u^{n_j}(\mu)]$
8:    end if
9: end for
```

examples studied in Section 8 based on our observation.

The ASS technique can be easily combined with the aforementioned algorithms for the generation of the RB and CRB. For example, Algorithm 4 shows the combination with the POD-Greedy algorithm (Algorithm 1). There is only one additional step in comparison with the original Algorithm 1.

Algorithm 4 RB generation using ASS-POD-Greedy

```
Input:    $\mathcal{P}_{\text{train}}$, $\mu_0$, $tol_{\text{RB}}(< 1)$
Output: RB $V = [V_1, \dots, V_N]$
1: Initialization: $N = 0$, $V = [\,]$, $\mu_{\max} = \mu_0$, $\eta(\mu_{\max}) = 1$
2: while $\eta_N(\mu_{\max}) > tol_{\text{RB}}$ do
3:    Compute the trajectory $S_{\max} := \{u^n(\mu_{\max})\}_{n=0}^{K}$ and adaptively select snapshots
      using Algorithm 3, and get
      $$S^A_{\max} := \{u^{n_1}(\mu_{\max}), \dots, u^{n_\ell}(\mu_{\max})\}.$$
4:    Enrich the RB, e.g. $V := [V, V_{N+1}]$, where $V_{N+1}$ is the first POD mode of
      the matrix $\bar{U}^A := [\bar{u}^{n_1}, \dots, \bar{u}^{n_\ell}]$ with $\bar{u}^{n_s} := u^{n_s}(\mu_{\max}) - \Pi_{\mathcal{W}^N}[u^{n_s}(\mu_{\max})]$, $s =$
      $1, \dots, \ell$, $\ell \ll K$. $\Pi_{\mathcal{W}^N}[u]$ is the projection of $u$ onto the current space $\mathcal{W}^N :=$
      $\text{span}\{V_1, \dots, V_N\}$.
5:    $N = N + 1$
6:    Find $\mu_{\max} := \arg\max_{\mu \in P_{\text{train}}} \eta_N(\mu)$.
7: end while
```

6 RB scheme for batch chromatography

Reduced basis methods are used to perform a rapid solution to the PDEs. The RBM based on FV-discretization for evolution equations is proposed in [24]. In this section, we show the derivation of the FOM based on the FV discretization for the batch chromatographic model (3)−(5), and the efficient generation of the ROM.

6.1 Full-order model based on FV discretization

As is mentioned in Section 1, we use the FV discretization for the batch chromatographic model (3)−(5). More specifically, we use the Lax-Friedrichs flux [28] for the convection contribution, central difference approximation for the diffusion terms, and the Crank-Nicolson scheme for the time discretization. The full discrete FV formulation for the system (3)−(4) can be written as follows

$$\begin{cases} Ac_z^{n+1} = Bc_z^n + d_z^n - \dfrac{1-\epsilon}{\epsilon}\Delta t h_z^n, \\ q_z^{n+1} = q_z^n + \Delta t h_z^n, \end{cases} \tag{12}$$

where $c_z^n := c_z^n(\mu) = ({c_z}_1^n, \dots, {c_z}_{\mathcal{N}}^n)^T, q_z^n := q_z^n(\mu) = ({q_z}_1^n, \dots, {q_z}_{\mathcal{N}}^n)^T \in \mathbb{R}^{\mathcal{N}}$, $z = \mathsf{a}, \mathsf{b}$, indicates the solutions of the field variables c_z and q_z at time instance $t = t^n\,(n = 0, \dots, K)$, A and B are tridiagonal constant matrices, d_z^n and h_z^n are parameter- and time-dependent,

$$d_z^n := d_0^n e_1, \quad h_z^n := ({h_z}_1^n, \dots, {h_z}_{\mathcal{N}}^n)^T,$$

with $d_0^n := \Delta x Pe\left(\frac{\lambda}{2} + \nu\right)\chi_{[0,t_{\text{in}}]}(t^n)$, $\lambda := \frac{\Delta t}{\Delta x}$, $\nu := \frac{\Delta t}{Pe\Delta x^2}$, $e_1 := (1, 0, \dots, 0)^T \in \mathbb{R}^{\mathcal{N}}$, and ${h_z}_j^n := h_z({c_\mathsf{a}}_j^n, {c_\mathsf{b}}_j^n, {q_z}_j^n) = \frac{L}{Q/(\epsilon A_c)}\kappa_z\left(f_z({c_\mathsf{a}}_j^n, {c_\mathsf{b}}_j^n) - {q_z}_j^n\right), j = 1, \dots, \mathcal{N}$.

6.2 Reduced-order model

Let $N \in \mathbb{N}^+$ be the number of the RB vectors for c_z and q_z, and $M \in \mathbb{N}^+$ be the number of the CRB vectors for the operators h_a and h_b. Here for simplicity of analysis, we use the same dimension N of the RB for $c_\mathsf{a}, c_\mathsf{b}, q_\mathsf{a}$ and q_b, but one can certainly take different dimensions for the RB. This also applies to h_a and h_b. Assume that $W_z \in \mathbb{R}^{\mathcal{N}\times M}$ is the CRB for the nonlinear operator h_z, and $V_{c_z}, V_{q_z} \in \mathbb{R}^{\mathcal{N}\times N}$ $\left(V_{c_z}^T V_{c_z} = I, V_{q_z}^T V_{q_z} = I\right)$ are the RB for the field variables c_z and q_z, respectively, i.e.,

$$h_z^n \approx W_z\beta_z^n, \quad c_z^n \approx \hat{c}_z^n := V_{c_z} a_{c_z}^n, \quad q_z^n \approx \hat{q}_z^n := V_{q_z} a_{q_z}^n, \quad n = 0, \dots, K. \tag{13}$$

Applying Galerkin projection and empirical operator interpolation, we formulate the ROM for the FOM (12) as follows,

$$\begin{cases} \hat{A}_{c_z} a_{c_z}^{n+1} = \hat{B}_{c_z} a_{c_z}^n + d_0^n \hat{d}_{c_z} - \dfrac{1-\epsilon}{\epsilon}\Delta t \hat{H}_{c_z}\beta_z^n, \\ a_{q_z}^{n+1} = a_{q_z}^n + \Delta t \hat{H}_{q_z}\beta_z^n, \end{cases} \tag{14}$$

where $a_{c_z}^n := a_{c_z}^n(\mu) = \left({a_{c_z}}_1^n, \dots, {a_{c_z}}_N^n\right)^T$, $a_{q_z}^n := a_{q_z}^n(\mu) = \left({a_{q_z}}_1^n, \dots, {a_{q_z}}_N^n\right)^T \in \mathbb{R}^N$ are the reduced state vector of the ROM. $\hat{A}_{c_z} = V_{c_z}^T A V_{c_z}, \hat{B}_{c_z} = V_{c_z}^T B V_{c_z}, \hat{d}_{c_z} = V_{c_z}^T e_1$, $\hat{H}_{c_z} := V_{c_z}^T W_z$, $\hat{H}_{q_z} := V_{q_z}^T W_z$ are the reduced matrices.

Note that $\beta_z^n := \beta_z^n(\mu) = ({\beta_z}_1^n, \dots, {\beta_z}_M^n)^T \in \mathbb{R}^M$ are the vectors of coefficients for the empirical interpolation of the nonlinear operator h_z^n, and are parameter- and time-dependent. The evaluation of β_z^n is essentially the same as the computation of the coefficients $\sigma_i(\mu)$ in (10) in Algorithm 2. More specifically, it can be obtained by solving the following system of equations:

$$\sum_{i=1}^{M} {\beta_z}_i^n W_{zi}(x_j) = {h_z}_j^n, \quad j = 1, \dots, M.$$

Here, the evaluation of ${h_z}_j^n$ only needs the j-th entries (${c_\mathsf{a}}_j^n, {c_\mathsf{b}}_j^n$ and ${q_z}_j^n$) of the solution vectors (${c_\mathsf{a}}^n, {c_\mathsf{b}}^n$ and ${q_z}^n$), i.e., ${h_z}_j^n = h_z({c_\mathsf{a}}_j^n, {c_\mathsf{b}}_j^n, {q_z}_j^n)$. For the general operator empirical interpolation, the value of the operator at the interpolation point (e.g. x_j) may depend on more entries of the solution vectors (e.g. the j-th entries and their neighbors). For more details, refer to [11, 25].

6.3 Offline-online decomposition

The efficiency of the RB approximation is ensured by a strategy of suitable offline-online decomposition which decouples the generation and projection of the RB approximation. Computation entails a possibly expensive offline phase performed only once and a cheap online phase for any chosen parameter in the parameter domain. During the offline stage, the RB, the CRB, reduced matrices and all $\mathcal{N}$-dependent terms are computed and stored; in the online process, for any given parameter μ, all parameter-dependent coefficients and the RB approximation are evaluated rapidly.

More precisely, in the offline process, given training sampling sets $\mathcal{P}_{\text{train}}^{\text{crb}}$ and $\mathcal{P}_{\text{train}}$ (they can be chosen differently), Algorithm 2 is implemented to generate the CRB W_z for the nonlinear operator h_z. Then Algorithm 4 is used to generate the reduced bases V_{c_z} and V_{q_z}. Consequently, all $\mathcal{N}$-dependent terms are precomputed and assembled to construct the reduced matrices (e.g. $\hat{A}_{c_z}, \hat{B}_{c_z}, \hat{d}_{c_z}, \hat{H}_{c_z}$, and $\hat{H}_{q_z}$), and the $\mathcal{N}$-independent ROM can be formulated as in (14). For a newly given parameter $\mu \in \mathcal{P}$, the low dimensional model (14) is solved online and the solution to the FOM (12) can be recovered by (13).

7 Output-oriented error estimation

It is crucial to get a sharp, rigorous and inexpensive posteriori error bound [34], which enables reliable and low-cost construction of the RB. In the past years, many efforts have been made for different problems, e.g. [11, 18, 24, 25, 35, 36]. One common technique for the derivation of the error estimator is based on the residual. In [11, 25], the authors provided an error estimation for the field variable in functional space for evolution equations. Since all the simulations are done in the finite dimensional vector space in practice, in this work, we derive an error estimation for the field variable directly in vector space, which is sharper than it is in the operator form. Moreover, we derive an output-oriented error bound based on the error estimation for the field variable. For many applications, the output response $y(u^{\mathcal{N}})$ is of interest. Hence, during the process of the greedy algorithm, e.g. Algorithm 1 or Algorithm 4, the error estimation $\eta_N(\mu_{\max})$ should be the error estimation for the output response, which is expected to be more accurate and reasonable.

In what follows, the inner product is defined as $\langle z_1, z_2\rangle := z_1^T z_2,\ \forall z_1, z_2 \in \mathbb{R}^{\mathcal{N}}$. The induced norm $\|\cdot\|$ is the standard 2-norm in the Euclidean space. However, if the discrete system of equations is obtained by using the finite element method, the solution to the discrete system is actually the coefficient vector corresponding to the basis vectors of the solution space. In such a case, the inner product should be defined properly with the mass matrix of the solution space, and the norm will be the induced norm correspondingly.

7.1 Output error estimation for the reduced order model

For a parametrized evolution equation (6), we derive an output error bound in the vector space for the ROM (9). Recall that in Section 3, $L_I(t^n)$ and $L_E(t^n)$ are linear, the evolution scheme (7) can be rewritten as follows in the vector space,

$$A^{(n)}u^{n+1}(\mu) = B^{(n)}u^n(\mu) + g\left(u^n(\mu), \mu\right), \tag{15}$$

where $A^{(n)}, B^{(n)} \in \mathbb{R}^{\mathcal{N}\times\mathcal{N}}$ are constant matrices and $g\left(u^n(\mu), \mu\right) \in \mathbb{R}^{\mathcal{N}}$ corresponds to the nonlinear term. Note that $A^{(n)}$ and $B^{(n)}$ are nonsingular for a stable scheme in practice, $n = 0, \dots, K-1$.

Given a parameter $\mu \in \mathcal{P}$, let $\hat{u}^n(\mu) = Va^n(\mu)$ be the RB approximation of $u^n(\mu)$, and $\hat{g}^n(\mu) := \mathcal{I}_M[g(\hat{u}^n(\mu)] = W\beta^n(\mu)$ be the interpolant of the nonlinear term, where $V \in \mathbb{R}^{\mathcal{N}\times N}, W \in \mathbb{R}^{\mathcal{N}\times M}$ are the precomputed parameter-independent bases, $a^n(\mu) \in \mathbb{R}^N$, $\beta^n(\mu) \in \mathbb{R}^M$ are parameter-dependent coefficients. In the following, for the sake of simplicity, we omit the explicit expression of the dependence on μ in $u^n(\mu), \hat{u}^n(\mu), a^n(\mu)$ and $\beta^n(\mu)$, and use $u^n, \hat{u}^n, a^n$ and β^n instead. The following a posteriori error estimation is based on the residual

$$r^{n+1}(\mu) := B^{(n)}\hat{u}^n + \mathcal{I}_M[g(\hat{u}^n)] - A^{(n)}\hat{u}^{n+1}. \tag{16}$$

With simple computation, we get the norm of the residual:

$$\begin{aligned}\left\|r^{n+1}(\mu)\right\|^2 =& \left\langle r^{n+1}(\mu), r^{n+1}(\mu)\right\rangle \\ =& (a^n)^T\, \underline{V^T(B^{(n)})^T B^{(n)}V}a^n + (\beta^n)^T\, \underline{W^T W}\beta^n \\ &+ \left(a^{n+1}\right)^T \underline{V^T(A^{(n)})^T A^{(n)}V}a^{n+1} + 2\left(\beta^n\right)^T \underline{W^T B^{(n)}V}a^n \\ &- 2(a^n)^T\underline{V^T(B^{(n)})^T A^{(n)}V}a^{n+1} - 2\left(\beta^n\right)^T \underline{W^T A^{(n)}V}a^{n+1}.\end{aligned} \tag{17}$$

Proposition 7.1. *Assume that the operator $g : \mathbb{R}^{\mathcal{N}} \to \mathbb{R}^{\mathcal{N}}$ is Lipschitz continuous, i.e., there exists a positive constant L_g, such that*

$$\|g(x) - g(y)\| \le L_g\|x - y\|, \quad x, y \in \mathcal{W}^{\mathcal{N}},$$

and the interpolation of g is "exact" with a certain dimension of $W = [W_1, \dots, W_{M+M'}]$, i.e.,

$$\mathcal{I}_{M+M'}\left[g(\hat{u}^n)\right] := \sum_{m=1}^{M+M'} W_m \cdot \beta_m^n = g(\hat{u}^n).$$

Assume again, that for all $\mu \in \mathcal{P}$, the initial projection error is vanishing $e^0(\mu) = 0$, then the approximation error $e^n(\mu) := u^n - \hat{u}^n$ satisfies

$$\|e^n(\mu)\| \leq \sum_{k=0}^{n-1} \left(\left\|(A^{(k)})^{-1}\right\| \prod_{j=k+1}^{n-1} \mathtt{G}^{(j)} \right) \left(\epsilon_{EI}^k(\mu) + \left\|r^{k+1}(\mu)\right\|\right), \tag{18}$$

where $\mathtt{G}^{(j)} = \left\|(A^{(j)})^{-1}\right\| \left(\|B^{(j)}\| + L_g\right)$, $\epsilon_{EI}^n(\mu)$ is the error due to the interpolation. A sharper error bound can be given as:

$$\begin{aligned} \|e^n(\mu)\| &\leq \eta_{N,M}^n(\mu) \\ &:= \sum_{k=0}^{n-1} \left(\prod_{j=k+1}^{n-1} \mathtt{G_F}^{(j)} \right) \left(\left\|(A^{(k)})^{-1}\right\| \epsilon_{EI}^k(\mu) + \left\|(A^{(k)})^{-1} r^{k+1}(\mu)\right\| \right), \end{aligned} \tag{19}$$

where $\mathtt{G_F}^{(j)} = \left\|(A^{(j)})^{-1}B^{(j)}\right\| + L_g \left\|(A^{(j)})^{-1}\right\|$, $n = 0, \dots, K-1$.

Proof. By forming the difference of (15) and (16), we have the error equation:

$$\begin{aligned} A^{(n)} e^{n+1}(\mu) &= B^{(n)} e^n(\mu) + g(u^n) - \mathcal{I}_M[g(\hat{u}^n)] + r^{n+1}(\mu) \\ &= B^{(n)} e^n(\mu) + (g(u^n) - g(\hat{u}^n)) + (g(\hat{u}^n) - \mathcal{I}_M[g(\hat{u}^n)]) + r^{n+1}(\mu). \end{aligned} \tag{20}$$

Multiplying by $(A^{(n)})^{-1}$ on both sides of (20), we obtain

$$\begin{aligned} e^{n+1}(\mu) =& (A^{(n)})^{-1} B^{(n)} e^n(\mu) + (A^{(n)})^{-1} \left(g(u^n) - g(\hat{u}^n)\right) \\ &+ (A^{(n)})^{-1} \left(g(\hat{u}^n) - \mathcal{I}_M\left[g(\hat{u}^n)\right]\right) + (A^{(n)})^{-1} r^{n+1}(\mu). \end{aligned} \tag{21}$$

Applying the Lipschitz condition of g, we have $\|g(u^n) - g(\hat{u}^n)\| \leq L_g \|e^n(\mu)\|$. Then by the triangle inequality and the property of the matrix norm, we have

$$\left\|e^{n+1}(\mu)\right\| \leq \left\|(A^{(n)})^{-1}\right\| \left(\left(\left\|B^{(n)}\right\| + L_g \right) \|e^n(\mu)\| + \epsilon_{EI}^n(\mu) + \left\|r^{n+1}(\mu)\right\| \right), \tag{22}$$

where $\epsilon_{EI}^n(\mu) := g(\hat{u}^n) - \mathcal{I}_M[g(\hat{u}^n)] = \sum_{m=M+1}^{M+M'} \|W_m\| \cdot |\beta_m^n|$. Resolving the recursion (22) with initial error $\left\|e^0(\mu)\right\| = 0$ yields the error bound in (18).

To get the error bound in (19), we re-observe the equation in (21), and see that the error bound in (22) is unnecessarily enlarged. A sharper bound for $\left\|e^{n+1}\right\|$ is of the following form,

$$\begin{aligned} \left\|e^{n+1}(\mu)\right\| \leq& \left(\left\|(A^{(n)})^{-1} B^{(n)}\right\| + L_g \left\|(A^{(n)})^{-1}\right\| \right) \|e^n(\mu)\| \\ &+ \left\|(A^{(n)})^{-1}\right\| \epsilon_{EI}^n(\mu) + \left\|(A^{(n)})^{-1} r^{n+1}(\mu)\right\|, \end{aligned} \tag{23}$$

since the following two inequalities are true, i.e., $\left\|(A^{(n)})^{-1}B^{(n)}\right\| \leq \left\|(A^{(n)})^{-1}\right\| \left\|B^{(n)}\right\|$ and $\left\|(A^{(n)})^{-1}r^{n+1}\right\| \leq \left\|(A^{(n)})^{-1}\right\| \left\|r^{n+1}\right\|$. Resolving the recursion (23) with initial error $\left\|e^0(\mu)\right\| = 0$ yields the proposed error bound in (19). □

Remark 7.2. In many cases, the operators $L_I(t^n)$ and $L_E(t^n)$ in (7) are independent of t^n, so that the coefficient matrices $A^{(n)}, B^{(n)}$ in (15) are constant matrices, see e.g. in (12). In such a case, the error bound becomes much more simple, see e.g. (31) and (33) in the next subsection.

Remark 7.3. In [11], the derivation of the error bound is based on the general operator form in the functional space. The error bound in (18) corresponds to the operator form (5.5) in [11]. However, the error bound may grow exponentially when $\mathtt{G}^{(j)} = \left\|(A^{(j)})^{-1}\right\| \left(\left\|B^{(j)}\right\| + L_g\right) > 1$ in (18). In the vector space, this problem can be easily avoided by using (23) instead of (22) if $\mathtt{G_F}^{(j)} = \left\|(A^{(n)})^{-1}B\right\| + L_g \left\|(A^{(n)})^{-1}\right\| \leq 1$, whereby the sharper error bound in (19) is obtained.

Remark 7.4. For the computation of the error bound in (18), we need to compute the norm of the residual $r^n(\mu)$ by using (17). Note that all terms underlined in (17) can be precomputed once V and W are obtained, and they are only computed once for all parameters in the training set. This is also true for the computation of $\left\|A^{-1}r^n(\mu)\right\|$ for the error bound in (19). Consequently, the evaluation of the error bound is cheap due to its independence of $\mathcal{N}$. In addition, as is shown in [11], small M' gives good results in practice; we use $M' = 1$ in the latter simulations.

Remark 7.5. Since the 2-norm is applied to the above error bound, and the 2-norm of a matrix H is actually its spectral norm. Therefore,

$$\left\|H^{-1}\right\| = \sigma_{\max}\left(H^{-1}\right) = \frac{1}{\sigma_{\min}(H)}.$$

As a result, the error bounds above are computable.

In many applications, the quantity of interest is not the field variable itself, but some outputs. In such cases, it is desired to estimate the output error to construct a ROM in a goal-oriented fashion. Based on the error estimation for the field variable above, we have the output error estimate below.

Proposition 7.6. *Under the assumptions of Proposition 7.1, assume the output of interest, $y\left(u^n(\mu)\right)$, can be expressed in the following form:*

$$y(u^n(\mu)) = Pu^n, \tag{24}$$

where $P \in \mathbb{R}^{N_o \times \mathcal{N}}$ is a constant matrix, then the output error $e_O^n(\mu) := Pu^n - P\hat{u}^n$ satisfies:

$$\begin{aligned} \left\|e_O^{n+1}(\mu)\right\| &\leq \tilde{\eta}_{N,M}^{n+1} \\ &:= \mathtt{G_O}^{(n)}\eta_{N,M}^n + \left\|P(A^{(n)})^{-1}\right\| \epsilon_{EI}^n(\mu) + \|P\| \left\|(A^{(n)})^{-1}r^{n+1}(\mu)\right\|, \end{aligned} \tag{25}$$

where $\mathtt{G_O}^{(n)} = \left\|P(A^{(n)})^{-1}B^{(n)}\right\| + L_g \left\|P(A^{(n)})^{-1}\right\|$, $n = 0, \ldots, K-1$.

Proof. Multiplying P from the left on both sides of the error equation (21), we get

$$\begin{aligned}Pe^{n+1}(\mu) = P\big((A^{(n)})^{-1}B^{(n)}e^{n}(\mu) + (A^{(n)})^{-1}\left(g(u^{n}) - g\left(\hat{u}^{n}\right)\right) \\ + (A^{(n)})^{-1}(g(\hat{u}^{n}) - \mathcal{I}_M[g(\hat{u}^{n})]) + (A^{(n)})^{-1}r^{n+1}(\mu)\big).\end{aligned}$$

Applying the Lipschitz condition of g, and using the triangle inequality, as well as the property of the matrix norm, we have:

$$\begin{aligned}\left\|e_O^{n+1}(\mu)\right\| &= \left\|Pe^{n+1}(\mu)\right\| \\ &\leq \mathtt{G_O}^{(n)}\left\|e^{n}(\mu)\right\| + \left\|P(A^{(n)})^{-1}\right\|\epsilon_{EI}^{n}(\mu) + \|P\|\left\|(A^{(n)})^{-1}r^{n+1}(\mu)\right\|.\end{aligned} \tag{26}$$

Replacing $\|e^n(\mu)\|$ in (26) with its bound in (19), we get the proposed output error bound in (25). □

Remark 7.7. Once the error estimation for the field variable is obtained, e.g. (19), a trivial error bound for the output (24) can be given as:

$$\begin{aligned}\left\|e_O^{n+1}(\mu)\right\| &= \left\|Pe^{n+1}(\mu)\right\| \\ &\leq \|P\|\left\|e^{n+1}(\mu)\right\| \\ &\leq \|P\|\left(\mathtt{G_F}^{(n)}\left\|e^{n}(\mu)\right\| + \left\|(A^{(n)})^{-1}\right\|\epsilon_{EI}^{n}(\mu) + \left\|(A^{(n)})^{-1}r^{n+1}(\mu)\right\|\right).\end{aligned} \tag{27}$$

The last inequality is true due to the inequality (23). It is obvious that the bound for $\left\|e_O^{n+1}(\mu)\right\|$ in (26) is sharper than that in (27). As a result, the final output error bound in (25) is sharper than the trivial output error bound derived in (27).

7.2 Output error estimation for the batch chromatographic model

The above error estimates are derived for a scalar evolution equation, a single PDE. For a system of several PDEs, one can analogously derive an error estimation for the underlying system by taking all the field variables as one vector. However, the behavior of the solution to each equation might be quite different, therefore, it is desired to generate different reduced bases for each field variable, rather than using a unified basis for all the field variables.

Here, we propose to derive an error estimation for each field variable for the underlying system (12) by following the error estimation in Section 7.1. Taking the error bound for the field variable c_z as an example, and recalling the detailed simulation for c_z (see (12)),

$$Ac_z^{n+1} = Bc_z^{n} + d_z^{n} - \frac{1-\epsilon}{\epsilon}\Delta t h_z^{n}, \tag{28}$$

the residual caused by the approximate solution $\hat{c}_z^n$ in (13) is

$$r_{c_z}^{n+1}(\mu) := B\hat{c}_z^{n} + d_z^{n} - \frac{1-\epsilon}{\epsilon}\Delta t\mathcal{I}_M[h_z(\hat{c}_z^{n})] - A\hat{c}_z^{n+1}. \tag{29}$$

Notice that the coefficient matrices A and B are independent of time, i.e., they are constant matrices. This indicates the following error bounds in (32) and (33) are relatively simple and cheap to compute.

Observe that (28), (29) correspond to (15) and (16) in the general case. Compared to the general form (15), the additional term d_z^n in (28) comes from the Neumann boundary condition, which does not depend on the solution c_z^n. Instead of requiring a Lipschitz continuity condition for h_z as a function of $c_{\mathtt{a}}^n$, $c_{\mathtt{b}}^n$ and q_z^n, we assume there exists a positive constant L_h such that,

$$\|h_z(c_{\mathtt{a}}^n, c_{\mathtt{b}}^n, q_z^n) - h_z(\hat{c}_{\mathtt{a}}^n, \hat{c}_{\mathtt{b}}^n, \hat{q}_z^n)\| \le L_h \|c_z^n - \hat{c}_z^n\|, \quad n = 0, \dots, K. \tag{30}$$

Assuming the initial projection error is vanishing $e_{c_z}^0(\mu) = 0$, we have a similar estimation for the approximation error $e_{c_z}^n(\mu) := c_z^n - \hat{c}_z^n$ $(n = 1, \dots, K)$ as follows,

$$\|e_{c_z}^n(\mu)\| \le \sum_{k=0}^{n-1} \|A^{-1}\|^{n-k} \mathtt{C}^{n-1-k} \left(\tau \epsilon_{EI}^k(\mu) + \|r_{c_z}^{k+1}(\mu)\|\right), \tag{31}$$

where $\mathtt{C} = \|B\| + \tau L_h$, $\tau = \frac{1-\epsilon}{\epsilon}\Delta t$. More tightly,

$$\begin{aligned} \|e_{c_z}^n(\mu)\| &\le \eta_{N,M,c_z}^n(\mu) \\ &:= \sum_{k=0}^{n-1} (\mathtt{G}_{\mathtt{F},\mathtt{c}})^{n-1-k} \left(\tau \|A^{-1}\| \epsilon_{EI}^k(\mu) + \|A^{-1} r_{c_z}^{k+1}(\mu)\|\right), \end{aligned} \tag{32}$$

where $\mathtt{G}_{\mathtt{F},\mathtt{c}} = \|A^{-1}B\| + \tau L_h \|A^{-1}\|$.

Analogously, the error bound for the output of interest $e_{c_z,O}^n(\mu) := P c_z^n - P\hat{c}_z^n$ can be obtained based on the error bound of the field variable. Similar to (25), we have

$$\begin{aligned} \left\|e_{c_z,O}^{n+1}(\mu)\right\| &\le \tilde{\eta}_{N,M,c_z}^{n+1}(\mu) \\ &:= \mathtt{G}_{\mathtt{O},\mathtt{c}} \eta_{N,M,c_z}^n(\mu) + \tau \|PA^{-1}\| \epsilon_{EI}^n(\mu) + \|P\| \|A^{-1} r_{c_z}^{n+1}(\mu)\|, \end{aligned} \tag{33}$$

where $\mathtt{G}_{\mathtt{O},\mathtt{c}} = \|PA^{-1}B\| + \tau L_h \|PA^{-1}\|$. Note that $P = (0, \dots, 0, 1) \in \mathbb{R}^{\mathcal{N}}$ in this model, which means the norm of the output $e_{c_z,O}^{n+1}(\mu)$ is the absolute value of the last entry of the error vector $e_{c_z}^{n+1}(\mu)$.

Remark 7.8. The error estimate for $q_{\mathtt{a}}$ and $q_{\mathtt{b}}$ in (12) can also be obtained similarly by following the derivation in Section 7.1. As the output of interest for the system in (12) only depends on $c_{\mathtt{a}}$ and $c_{\mathtt{b}}$, the error estimates for q_a and q_b are not needed for the output error bound, and therefore will not be presented here.

Remark 7.9. As is mentioned above, it is possible to derive an error estimation for the field variables $\mathbf{U} = (c_{\mathtt{a}}, c_{\mathtt{b}}, q_{\mathtt{a}}, q_{\mathtt{b}})^T$ by considering $h_z(c_{\mathtt{a}}, c_{\mathtt{b}}, q_z)$ as a function of the vector $\mathbf{U}$. However, for the output error bound in (33), the error bound $\eta_{N,M,c_z}^n(\mu)$ for the field variable c_z is involved, so is the desired error bound (denoted as $\eta_{N,M,\mathbf{U}}^n(\mu)$) for the vector $\mathbf{U}$, if the output error estimation is derived by considering all the field variables together. Obviously, the error bound $\eta^n_{N,M,\mathbf{U}}(\mu)$ is much rougher than the bound $\eta^n_{N,M,c_z}(\mu)$.

The assumption (30) is easily fulfilled in practice. In fact, the constant L_h can be conservatively chosen large, and the weight τL_h is still small because the time step Δt is typically very small.

7.3 An early-stop criterion for the Greedy algorithm

From the expression of the error estimator above, it is seen that the error bound for the field variables ($\eta^n_{N,M}(\mu)$ or $\eta^n_{N,M,c_z}(\mu)$) is accumulated with time. Since $\eta^n_{N,M}(\mu)$ (or $\eta^n_{N,M,c_z}(\mu)$, respectively) is involved in the output error bound in (25) (or (33), respectively), the output error bound is also accumulated with time. As a result, the output error bound at the final time steps may not reflect the true error after a long evolution process. Figure 2 in Section 8.2 illustrates this behavior. In fact, similar phenomena are reported in the literature, e.g. [30], where it is pointed out that the error estimate, e.g. in (18), may loose sharpness when many time instances $t^n, n = 0, 1, \ldots, K$, are needed for a given time interval $[0, T]$, which is typical for convection-diffusion equations with small diffusion terms. However, the output error bound is cheap to compute, and it may still provide a guidance for the parameter selection in the greedy algorithm.

To circumvent the problem above, we add a validation step to get an early-stop of the extension process, as is shown in Algorithm 5. More precisely, after Step 6 in Algorithm 4, we compute the decay rate $d\eta$ of the error bound. If $d\eta$ is smaller than a predefined tolerance, indicating the error bound stagnates, then we further check the true output error at the parameter $\mu_{\max}$ determined by the greedy algorithm. When the true output error at $\mu_{\max}$ is smaller than tol_{RB}, we assume that there is no need to include a new basis vector, and the RB extension can be stopped; otherwise, the process should continue.

Algorithm 5 RB generation using ASS-POD-Greedy with early-stop

```
Input:   $\mathcal{P}_{\mathrm{train}}$, $\mu_0$, $tol_{\mathrm{RB}}(< 1)$, $tol_{\mathrm{decay}}$
Output:  RB $V = [V_1, \ldots, V_N]$
 1: Implement Step 1 in Algorithm 4.
 2: while the error $\eta_N(\mu_{\max}) > tol_{\mathrm{RB}}$ do
 3:    Implement Steps 3–6 in Algorithm 4.
 4:    Compute the decay rate of the error bound $d\eta = \frac{\eta_{N-1}(\mu^{\mathrm{old}}_{\max}) - \eta_N(\mu_{\max})}{\eta_{N-1}(\mu^{\mathrm{old}}_{\max})}$.
 5:    if $d\eta < tol_{\mathrm{decay}}$ then
 6:       Compute the true output error at the selected parameter $\mu_{\max}$, $\bar{e}_N(\mu_{\max})$.
 7:       if $\bar{e}_N(\mu_{\max}) < tol_{\mathrm{RB}}$ then
 8:          Stop
 9:       end if
10:    end if
11: end while
```

Remark 7.10. It can happen that the error bound stagnates for a while, but then again decays. In order to monitor such a case, the tolerance tol_{decay} should be set to a very small value, which allows some steps of stagnation. If after some steps of stagnation, the error bound still does not decay, then Step 5 in Algorithm 5 will be implemented.

8 Numerical experiments

In this work, the RB methodology is employed to accelerate the optimization with nonlinear PDE constraints. As a case study, we investigate the optimal operation of batch chromatography. More specifically, the operating variable $\mu = (Q, t_{\text{in}})$ is optimally chosen in a reasonable parameter domain to maximize the production rate $Pr(\mu) =: \frac{s(\mu)Q}{t_{cyc}}$, while respecting the requirement of the recovery yield $Rec(\mu) := \frac{s(\mu)}{t_{\text{in}}(c_{\mathsf{a}}^{\text{f}}+c_{\mathsf{b}}^{\text{f}})}$. Here, $s(\mu) := \int_{t_3}^{t_4} c_{\mathsf{a},O}(t;\mu)dt + \int_{t_1}^{t_2} c_{\mathsf{b},O}(t;\mu)dt$, $c_{z,O}(t;\mu) := c_z(t,1;\mu)$, is the concentration of component z ($z = \mathsf{a}, \mathsf{b}$) at the outlet of the column. We consider the optimization problem of batch chromatography as follows,

$$\begin{aligned} &\min_{\mu \in \mathcal{P}} \{-Pr(\mu)\}, \\ s.t.\ &Rec_{\text{min}} - Rec(\mu) \leq 0, \quad \mu \in \mathcal{P} \\ &c_z(\mu), q_z(\mu) \text{ are the solutions to the system } (3)-(5), z = \mathsf{a}, \mathsf{b}. \end{aligned} \tag{34}$$

Notice that when solving the system (3)−(5), the time step size has to be taken relatively small so that the cutting points t_i can be determined properly, $i = 1, \ldots, 4$, and the the integral in $s(\mu)$ can be evaluated accurately. The small step size results in a very large number (up to $O(10^4)$) of total time steps for every parameter $\mu \in \mathcal{P}$, which causes a lot of difficulties in the error estimation and the generation of the reduced basis.

The model parameters and operating conditions are presented in Table 1. The Henry constants and thermodynamic coefficients in the isotherm equation (4) are given in Table 2. The parameter domain for the operating variable μ is $\mathcal{P} = [0.0667, 0.1667] \times [0.5, 2.0]$. The minimum recovery yield Rec_{min} is taken as 80.0%, and the purity requirements are specified as $Pu_{\mathsf{a}} = 95.0\%, Pu_{\mathsf{b}} = 95.0\%$, which determine the cutting points t_2 and t_3 in $s(\mu)$. To capture the dynamics precisely, the dimension of spatial discretization $\mathcal{N}$ in the FOM (12) is taken as 1500.

Table 1: Model parameters and operating conditions for the chromatographic model.

Column dimensions [cm]	2.6×10.5
Column porosity ϵ [-]	0.4
Péclet number Pe [-]	2000
Mass-transfer coefficients κ_z, $z = \mathsf{a}, \mathsf{b}$ [1/s]	0.1
Feed concentrations c_z^{f}, $z = \mathsf{a}, \mathsf{b}$ [g/l]	2.9

Table 2: Coefficients of the adsorption isotherm equation.

$H_{\mathsf{a}1}$ [-]	2.69	$H_{\mathsf{b}1}$ [-]	3.73
$H_{\mathsf{a}2}$ [-]	0.1	$H_{\mathsf{b}2}$ [-]	0.3
$K_{\mathsf{a}1}$ [l/g]	0.0336	$K_{\mathsf{b}1}$ [l/g]	0.0446
$K_{\mathsf{a}2}$ [l/g]	1.0	$K_{\mathsf{b}2}$ [l/g]	3.0

In this section, we first illustrate the performance of the adaptive snapshot selection for the generation of the RB and CRB, and then show the output error estimation for the generation of the RB. Finally we present the results for the ROM-based optimization of batch chromatography. All the computations were done on a PC with Intel(R) Core(TM)2 Quad CPU Q9550 2.83GHz RAM 4.00GB unless stated otherwise.

8.1 Performance of the adaptive snapshot selection

To investigate the performance of the technique of adaptive snapshot selection, we compare the runtime for the generation of the RB and CRB with different threshold values tol_{ASS}. As is shown in Algorithm 4 in Section 5, the ASS can be combined with the POD-Greedy algorithm and used for the generation of the RB. For the computation of the error indicator $\eta_N(\mu_{\text{max}})$ in Algorithm 4, EI is involved for an efficient offline-online decomposition. To efficiently generate a CRB, the ASS is also employed. The training set for the generation of the CRB is a sample set with 25 sample points of $\mu = (Q, t_{\text{in}})$, uniformly distributed in the parameter domain. For each sample point, Algorithm 3 is used to adaptively choose the snapshots for the generation of the CRB. The runtimes for the CRB generation with different choices of tol_{ASS} are shown in Table 3. It is seen that, the larger threshold is used, the more runtime is saved. This means that a lot of redundant information is discarded due to the adaptive selection process. Particularly, with the tolerance $tol_{\text{ASS}} = 1.0 \times 10^{-4}$, the computational time is reduced by 90.3% compared to that of the original algorithm without ASS. However, how to choose an optimal threshold is empirical and problem-dependent.

Table 3: Illustration of the generation of CRBs (W_{a}, W_{b}) at the same error tolerance ($tol_{\text{CRB}} = 1.0 \times 10^{-7}$) with different thresholds. $M' = 1$, is the number of the basis for error estimation.

	tol_{ASS}	Res.($\|\xi^{\mathsf{a}}_{M+M'}\|$	$\|\xi^{\mathsf{b}}_{M+M'}\|$)	M (W_{a}	W_{b})	Runtime [h]
no ASS	–	9.2×10^{-8}	8.5×10^{-8}	146	152	62.5 (-)
ASS	1.0×10^{-4}	9.6×10^{-8}	8.1×10^{-8}	147	152	6.05 (-90.3%)
ASS	1.0×10^{-3}	8.7×10^{-8}	9.9×10^{-8}	147	152	3.62(-94.2%)
ASS	1.0×10^{-2}	9.4×10^{-8}	6.2×10^{-8}	144	150	2.70 (-95.7%)

Table 4 shows the comparison of the runtime for the generation of the RB by using the POD-Greedy algorithm with and without ASS. Note that the CRB is precomputed with $tol_{\mathrm{ASS}} = 1.0 \times 10^{-4}$ for the ASS, and the corresponding runtime is not included here. The training set is a sample set with 60 points uniformly distributed in the parameter domain. Here and in the following, the tolerances are chosen as $tol_{\mathrm{CRB}} = 1.0 \times 10^{-7}, tol_{\mathrm{RB}} = 1.0 \times 10^{-6}, tol_{\mathrm{ASS}} = 1.0 \times 10^{-4}$. It is seen that the runtime for generating the ROM with the ASS is reduced by 51.2% compared to that without ASS at the same tolerance tol_{RB}. Moreover, the accuracy of the resulting reduced model with ASS is almost the same as that without ASS, as is shown in Table 5.

Table 4: Comparison of the runtime for the RB generation using the POD-Greedy algorithm with early-stop (Algorithm 5) with and without ASS.

Algorithms	Runtime [h]
POD-Greedy	16.22 [1]
ASS-POD-Greedy	7.92 (−51.2%)

[1] Due to the memory limitation of the PC, the computation was done on a Workstation with 4 Intel Xeon E7-8837 CPUs (8 cores per CPU) 2.67 GHz RAM 1TB.

8.2 Performance of the output error bound

As aforementioned, it is wise to use an efficient error estimation for the output for the generation of the RB. In the chromatographic model, given a parameter μ, the values of $Pr(\mu)$ and $Rec(\mu)$ in (34) are determined by the concentrations at the outlet of the column $c^n_{z,O}(\mu) = Pc^n_z(\mu), n = 0, \dots, K$, $z = \mathsf{a}, \mathsf{b}$, which constitute the output of the FOM in (12). Consequently, the output error bound will be taken as the error indicator $\eta_N(\mu)$ in the greedy algorithm (e.g. Algorithm 4, 5) for the generation of the RB, which yields a goal-oriented ROM.

Note that the error bound $\tilde{\eta}^{n+1}_{N,M,c_z}(\mu)$ in (33) is the bound for the output error of the component c_z at the time instance t^{n+1} for a given parameter $\mu \in \mathcal{P}$. We use the following error bound in Algorithm 4, $\eta_N(\mu_{\max}) := \max\limits_{\mu \in \mathcal{P}_{\mathrm{train}}} \max\limits_{z \in \{\mathsf{a},\mathsf{b}\}} \bar{\tilde{\eta}}_{N,M,c_z}(\mu)$, where $\bar{\tilde{\eta}}_{N,M,c_z}(\mu) := \frac{1}{K}\sum_{n=1}^{K} \tilde{\eta}^n_{N,M,c_z}(\mu)$ is the average of the error bound for the output of c_z in the whole evolution process. In accordance, we define the reference true output error as $e^{\max}_N := \max\limits_{\mu \in \mathcal{P}_{\mathrm{train}}} \bar{e}_N(\mu)$, where $\bar{e}_N(\mu) := \max\limits_{z \in \{\mathsf{a},\mathsf{b}\}} \bar{e}_{N,c_z}(\mu)$, $\bar{e}_{N,c_z}(\mu) := \frac{1}{K}\sum_{n=1}^{K} \|c^n_{z,O}(\mu) - \hat{c}^n_{z,O}(\mu)\|$, and $\hat{c}^n_{z,O}(\mu)$ is the approximate output response computed from the ROM in (14).

Figure 2 shows the error decay and the true error as the RB is extended by using Algorithm 4. It can be seen that the output error bound stagnates after certain steps, although the true error is very small already. To circumvent the problem, Algorithm 5 is implemented to get an early-stop.

Figure 3 shows the results for Algorithm 5, where $tol_{\mathrm{ASS}} = 0.03$. Using the early-stop, the greedy algorithm can be terminated in time and the dimension of the RB can be kept small without loosing accuracy. It can be seen that 47 RB vectors already make the true output error very small, and the output error bound begins to stagnate, so that the early-stop gives a reasonable stopping criterion. Figure 4 shows the parameters selected in the RB extension with the greedy algorithm. The size of the circle shows how frequently the same parameter is selected for the RB extension.

Here, we want to point out that the difference between the error bound for the field variable and the output error bound is not so big. This is not surprising, because the derivation of the error bound for the output is based on that of the field variable. The technique of using the dual system could be employed to improve the error estimates, which will be investigated in the future.

Finally, to further assess the reliability and efficiency of the ROM, we performed the detailed and reduced simulation using a validation set $\mathcal{P}_{val}$ with 600 random sample points in the parameter domain. Table 5 shows the average runtime over the validation set, and the maximal error defined as Max.error $= \max_{\mu \in \mathcal{P}_{val}} \bar{e}_N(\mu)$. It is seen that the average runtime for a detailed simulation is sped up by a factor of 53, and the maximal true error is below the prespecified tolerance tol_{RB}. In addition, the concentrations at the outlet of the column computed by using the FOM and the ROM at a given parameter $\mu = (Q, t_{\mathrm{in}}) = (0.1018, 1.3487)$ are plotted in Figure 5, which shows that the ROM (14) reproduced the dynamics of the original FOM (12).

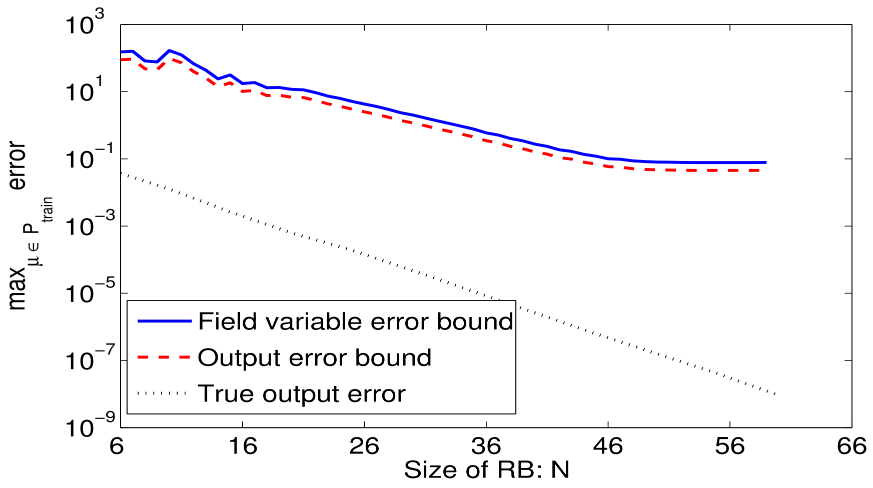

Figure 2: Illustration of the error bound decay during the RB extension using Algorithm 4, and the corresponding true output error. The output error bound $\eta_N(\mu_{\max})$ and the maximal true output error $e_N^{\max}$ are defined in Section 8.2, the field variable error bound is defined as $\eta_N := \max_{\mu \in \mathcal{P}_{\mathrm{train}}} \max_{z \in \{\mathsf{a},\mathsf{b}\}} \{\bar{\eta}_{N,M,c_z}(\mu)\}$, where $\bar{\eta}_{N,M,c_z}(\mu) := \frac{1}{K} \sum_{n=1}^{K} \eta_{N,M,c_z}^n(\mu)$.

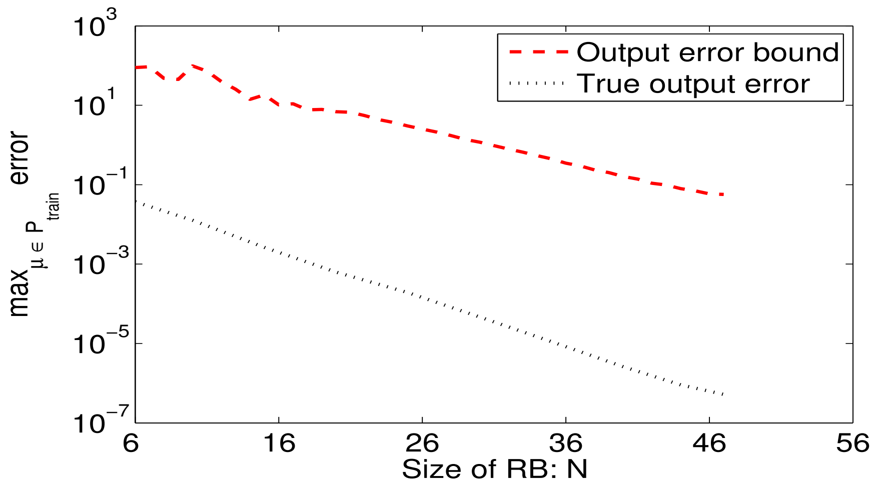

Figure 3: Error bound decay during the RB extension using the early-stop technique, Algorithm 5 and the corresponding maximal true output error.

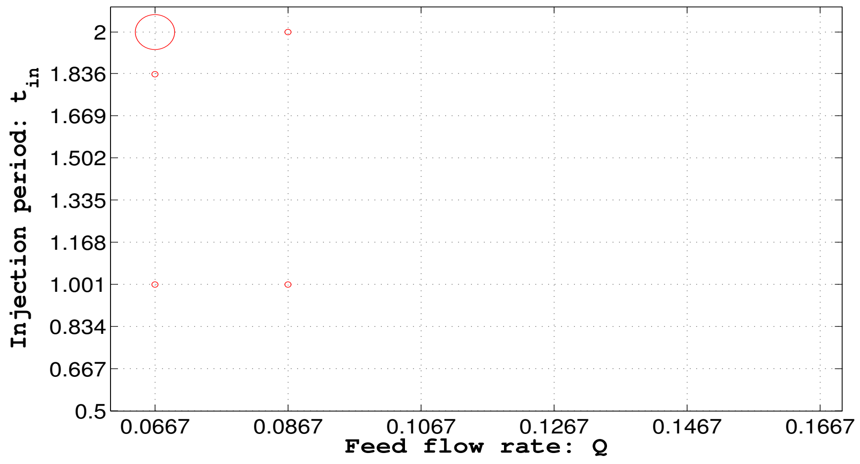

Figure 4: Parameter selection in the generation of the RB. The size of a circle indicates how frequently the parameter is selected during the process. The bigger the circle, the more often the parameter is selected.

Table 5: Runtime comparison of the detailed and reduced simulations over a validation set P_{val} with 600 random sample points. Tolerances for the generation of the ROM: $tol_{\mathrm{CRB}} = 1.0 \times 10^{-7}, tol_{\mathrm{ASS}} = 1.0 \times 10^{-4}, tol_{\mathrm{RB}} = 1.0 \times 10^{-6}$.

Simulations	Max. error	Average runtime [s]/SpF
FOM ($\mathcal{N} = 1500$)	–	312.13(-)
ROM, POD-Greedy	3.79×10^{-7}	6.3 / 50
ROM, ASS-POD-Greedy	4.58×10^{-7}	6.3 / 50

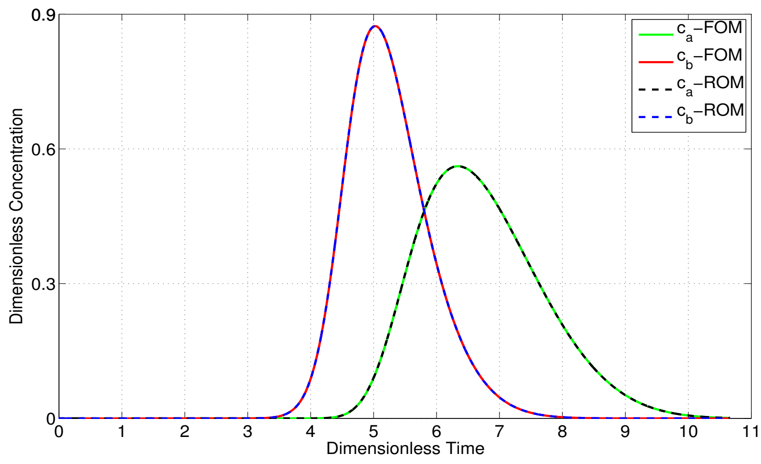

Figure 5: Concentrations at the outlet of the column using the FOM ($\mathcal{N} = 1500$) and the ROM ($N = 47$) at the parameter $\mu = (Q, t_{\mathrm{in}}) = (0.1018, 1.3487)$.

8.3 ROM-based Optimization

Once the ROM (14) is obtained, the original FOM-based optimization problem (34) can be approximated by the following ROM-based optimization problem:

$$\begin{aligned} &\min_{\mu \in \mathcal{P}} \{-\hat{P}r(\mu)\}, \\ s.t.\ &Rec_{\min} - \hat{R}ec(\mu) \leq 0, \\ &\hat{c}_z^n(\mu), \hat{q}_z^n(\mu) \text{ are the RB approximations from the ROM (14)}, z = \mathsf{a}, \mathsf{b}. \end{aligned}$$

Here $\hat{P}r(\mu)$ and $\hat{R}ec(\mu)$ are the approximated production and recovery yield, respectively. More specifically, at each iteration of the optimization process, for a selected parameter μ, the production and recovery yield are evaluated by solving the ROM (14) rather than the original FOM (12).

In this work, we use the global optimizer NLOPT_GN_DIRECT_L, an efficient gradient–free algorithm in the open library NLopt [27], to solve the optimization problems. Let μ^k be the vector of parameters determined by the optimization procedure at the kth iteration, $k = 1, 2, \ldots$. When $\|\mu^{k+1} - \mu^k\| < tol_{\text{opt}}$, the optimization process is stopped and the optimal solution is obtained. The tolerance is specified as $tol_{\text{opt}} = 1.0 \times 10^{-4}$. The optimization results are shown in Table 8.3. The optimal solution to the ROM-based optimization converges to that of the FOM-based one. Furthermore, the runtime for solving the FOM-based optimization is significantly reduced. The speed-up factor (SpF) is 54.

Note that if the offline cost, i.e., the runtime for constructing the ROM, is taken into account, the total cost of solving the ROM-based optimization is 79.35 hours if there is no ASS for the generation of the CRB and RB. It is even longer than the runtime for directly solving the FOM-based optimization. The latter is just 33.88 hours. However, when the technique of ASS was implemented for the construction of the ROM, the total cost of solving the ROM-based optimization is only 14.6 hours, which is less than half of the runtime for solving the FOM-based one. Needless to say, the gain is much more when the ROM is repeatedly used for multi-query context.

In fact, although the offline cost is usually not considered in the RB community, the total cost is an issue for many applications, e.g. the ROM-based optimization in this paper. For many simulations with varying parameters, the following two runtimes should be well balanced: one is constructing and using a surrogate ROM; the other is directly using the original FOM.

Table 6: Comparison of the optimization based on the ROM and FOM.

Simulations	Objective (Pr)	Opt. solution (μ)	N_it [1]	Runtime [h]/SpF
FOM-based Opt.	0.020264	$(0.07964, 1.05445)$	202	33.88 / -
ROM-based Opt.	0.020266	$(0.07964, 1.05445)$	202	0.63 / 54

[1] N_it denotes the number of iterations required to converge.

9 Conclusions

We have discussed how to use a surrogate ROM to solve an optimization problem constrained by parameterized PDEs with nonlinearity, and applied it to batch chromatography.

As a robust PMOR method, the RBM serves to generate the ROM. The empirical operator interpolation has been employed for an efficient offline-online decomposition. The ASS technique is proposed for an efficient generation of the RB and/or CRB, by which the offline time was significantly reduced with negligible loss of accuracy. An output-oriented error estimation is derived based on the residual in the vector space. However, the output error bound is conservative due to the error accumulation with time evolution. To circumvent the stagnation of the error bound, an early-stop criterion was proposed to make the RB extension stop in time with a desired accuracy. The resulting goal-oriented ROM is reliable and efficient over the whole parameter domain, and is qualified for optimization. To avoid the error accumulation in the error bound, output error estimation using the dual system should be a good candidate, and is under current investigation.

References

[1] Z. Bai, *Krylov subspace techniques for reduced-order modeling of large-scale dynamical systems*, Applied Numerical Mathematics, 43 (2002), pp. 9–44.

[2] M. Barrault, Y. Maday, N. C. Nguyen, and A. T. Patera, *An 'empirical interpolation' method: application to efficient reduced-basis discretization of partial differential equations*, Comptes Rendus Mathematique Academy Science Paris Series I, 339 (2004), pp. 667–672.

[3] U. Baur, C. Beattie, P. Benner, and S. Gugercin, *Interpolatory projection methods for parameterized model reduction*, SIAM Journal on Scientific Computing, 33 (2011), pp. 2489–2518.

[4] P. Benner, L. Feng, S. Li, and Y. zhang, *Reduced-order modeling and rom-based optimization of batch chromatography*, in ENUMATH 2013 Proceedings, accepted.

[5] P. Benner, S. Gugercin, and K. Willcox, *A survey of model reduction methods for parametric systems*, MPI Magdeburg Preprints, (2013).

[6] P. Benner, E. Sachs, and S. Volkwein, *Model Order Reduction for PDE Constrained Optimization*, Preprints Konstanzer Online-Publikations-System (KOPS), (2014).

[7] L. T. Biegler, O. Ghattas, M. Heinkenschloss, D. Keyes, and B. van Bloemen Waanders, *Real-Time PDE-constrained Optimization*, Society for Industrial and Applied Mathematics, 2007.

[8] L. T. Biegler, O. Ghattas, M. Heinkenschloss, and B. van Bloemen Waanders, *Large-scale PDE-constrained Optimization*, Springer, 2003.

[9] T. Bui-Thanh, *Model-constrained optimization methods for reduction of parameterized large-scale systems*, PhD thesis, Massachusetts Institute of Technology, 2007.

[10] T. Bui-Thanh, K. Willcox, and O. Ghattas, *Model reduction for large-scale systems with high-dimensional parametric input space*, SIAM Journal on Scientific Computing, 30 (2008), pp. 3270–3288.

[11] M. Drohmann, B. Haasdonk, and M. Ohlberger, *Reduced basis approximation for nonlinear parametrized evolution equations based on empirical operator interpolation*, SIAM Journal on Scientific Computing, 34 (2012), pp. 937–969.

[12] J. L. Eftang, D. J. Knezevic, and A. T. Patera, *An hp certified reduced basis method for parametrized parabolic partial differential equations*, Mathematical and Computer Modelling of Dynamical Systems, 17 (2011), pp. 395–422.

[13] M. Fahl, *Trust-region methods for flow control based on reduced order modelling*, PhD thesis, Universität Trier, 2000.

[14] M. Fahl and E. W. Sachs, *Reduced order modelling approaches to PDE-constrained optimization based on proper orthogonal decomposition*, in Large-scale PDE-constrained optimization, Springer, 2003, pp. 268–280.

[15] L. Feng and P. Benner, *A robust algorithm for parametric model order reduction*, Proceedings in Applied Mathematics and Mechanics, 7 (2008), pp. 10215.01–10215.02.

[16] L. Feng, P. Benner, and J. G. Korvink, *Subspace recycling accelerates the parametric macro-modeling of MEMS*, International Journal for Numerical Methods in Engineering, 94 (2013), pp. 84–110.

[17] W. Gao and S. Engell, *Iterative set-point optimization of batch chromatography*, Computers & Chemical Engineering, 29 (2005), pp. 1401–1409.

[18] M. A. Grepl, *Reduced-basis approximation a posteriori error estimation for parabolic partial differential equations*, PhD thesis, Massachusetts Institute of Technology, 2005.

[19] D. Gromov, S. Li, and J. Raisch, *A hierarchical approach to optimal control of a hybrid chromatographic batch process*, in Advanced Control of Chemical Processes, vol. 7, 2009, pp. 339–344.

[20] G. Guiochon, A. Felinger, D. G. Shirazi, and A. M. Katti, *Fundamentals of Preparative and Nonlinear Chromatography*, Academic Press, 2006.

[21] B. Haasdonk, *Convergence rates of the POD-Greedy method*, ESAIM: Mathematical Modelling and Numerical Analysis, 47 (2013), pp. 859–873.

[22] B. Haasdonk, M. Dihlmann, and M. Ohlberger, *A training set and multiple bases generation approach for parameterized model reduction based on adaptive grids in parameter space*, Mathematical and Computer Modelling of Dynamical Systems, 17 (2011), pp. 423–442.

[23] B. Haasdonk and M. Ohlberger, *Adaptive basis enrichment for the reduced basis method applied to finite volume schemes*, in Proc. 5th International Symposium on Finite Volumes for Complex Applications, 2008, pp. 471–478.

[24] ——, *Reduced basis method for finite volume approximations of parametrized linear evolution equations*, ESAIM: Mathematical Modelling and Numerical Analysis, 42 (2008), pp. 277–302.

[25] B. Haasdonk, M. Ohlberger, and G. Rozza, *A reduced basis method for evolution schemes with parameter-dependent explicit operators*, Electronic Transactions on Numerical Analysis, 32 (2008), pp. 145–161.

[26] K.-H. Hoffmann, G. Leugering, and F. Tröltzsch, *Optimal control of partial differential equations: international conference in Chemnitz, Germany, April 20-25, 1998*, vol. 133, Springer, 1999.

[27] S. G. Johnson, *The NLopt nonlinear-optimization package*, http://ab-initio.mit.edu/nlopt, (2010).

[28] R. J. LeVeque, *Finite volume methods for hyperbolic problems*, vol. 31, Cambridge University Press, 2002.

[29] A. Manzoni, A. Quarteroni, and G. Rozza, *Shape optimization for viscous flows by reduced basis methods and free-form deformation*, International Journal for Numerical Methods in Fluids, 70 (2012), pp. 646–670.

[30] N.-C. Nguyen, G. Rozza, and A. T. Patera, *Reduced basis approximation and a posteriori error estimation for the time-dependent viscous Burgers' equation*, Calcolo, 46 (2009), pp. 157–185.

[31] A. K. Noor and J. M. Peters, *Reduced basis technique for nonlinear analysis of structures*, AIAA Journal, 18 (1980), pp. 145–161.

[32] M. Ohlberger and M. Schaefer, *Reduced basis method for parameter optimization of multiscale problems*, in Proceedings of ALGORITMY 2012, 2012, pp. 1–10.

[33] I. B. Oliveira and A. T. Patera, *Reduced-basis techniques for rapid reliable optimization of systems described by affinely parametrized coercive elliptic partial differential equations*, Optimization and Engineering, 8 (2007), pp. 43–65.

[34] A. T. Patera and G. Rozza, *Reduced basis approximation and a posteriori error estimation for parametrized partial differential equations*, Version 1.0, Copyright MIT 2006.

[35] C. Prud'homme, D. V. Rovas, K. Veroy, L. Machiels, Y. Maday, A. T. Patera, and G. Turinici, *Reliable real-time solution of parametrized partial differential equations: Reduced-basis output bound methods*, Journal of Fluids Engineering, 124 (2002), pp. 70–80.

[36] D. V. Rovas, *Reduced-basis output bound methods for parametrized partial differential equations*, PhD thesis, Massachusetts Institute of Technology, 2003.