MedRed, 2013
Magdeburg, Dec 11-13

A reduced basis method and ROM-based optimization for batch chromatography

Yongjin Zhang

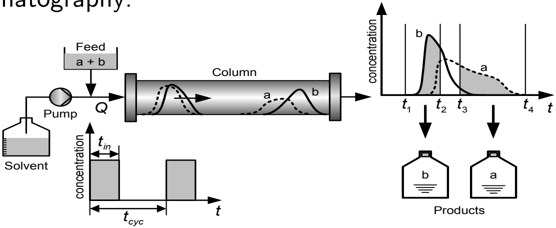
joint work with Peter Benner, Lihong Feng and Suzhou Li

Max Planck Institute for Dynamics of Complex Technical Systems
Magdeburg, Germany

Outline

Motivation : Model Description

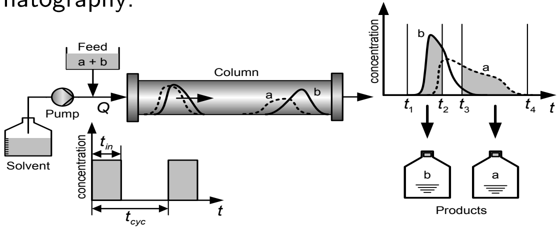
Batch chromatography:



Principle of batch chromatography for binary separation.

Motivation : Model Description

Batch chromatography:



Principle of batch chromatography for binary separation.

$$Eqs: \begin{cases} \frac{\partial c_z}{\partial t} + \frac{1-\epsilon}{\epsilon}\frac{\partial q_z}{\partial t} = -\frac{\partial c_z}{\partial x} + \frac{1}{\mathbf{Pe}}\frac{\partial^2 c_z}{\partial x^2}, & 0 < x < 1, \\ \frac{\partial q_z}{\partial t} = \frac{L}{Q/(\epsilon A_c)}\kappa_z(q_z^{Eq} - q_z), & 0 \le x \le 1, \end{cases} \quad (1)$$

with suitable initial and boundary conditions.
$q_z^{Eq} = f_z(c_{\mathrm{a}}(\mu), c_{\mathrm{b}}(\mu))$ is nonlinear and nonaffine, $z = \mathrm{a}, \mathrm{b}$.
$\mu := (Q, t_{in})$ is the vector of operating parameters.
$\mathbf{Pe} = 2000$.
Ref: *http://www.modelreduction.org*

Motivation : Model Description (cont.)

The optimization of batch chromatography:

$$\min_{\mu \in \mathcal{P}} \{-Pr(c_z(\mu), q_z(\mu); \mu)\}$$

$$s.t. \quad Rec(c_z(\mu), q_z(\mu); \mu) \geq Rec_{min},$$

$c_z(\mu), q_z(\mu)$ are the solutions to the above system (1).

Production rate: $Pr = \frac{p(\mu)Q}{t_{cyc}}$

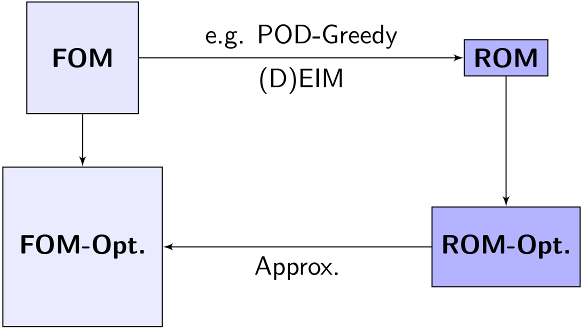
Recovery yield: $Rec = \frac{p(\mu)}{t_{in}(c_{\mathrm{a}}^f + c_{\mathrm{b}}^f)}$

$p(\mu) := \int_{t_1}^{t_2} c_{\mathrm{b},O}(t;\mu)dt + \int_{t_3}^{t_4} c_{\mathrm{a},O}(t;\mu)dt$

$c_{z,O}(t;\mu) := c_z(t, x = 1; \mu)$, $t \in [0, T]$: concentrations at the outlet

Motivation : General Idea

RBM & ROM-based optimization:



- Certified error estimation
- Offline-online decomposition
- Efficiency of the ROM (-based optimization)

Review on RBM

Consider a spatial discretized parametrized time–dependent system

$$\Sigma : \quad F(u(t;\mu)) = 0, \mu \in \mathcal{P}, t \in [0, T],$$

where $u(t;\mu) \in \mathbb{R}^{\mathcal{N}}$ and F is a linear or nonlinear operator. The output of interest $y := y(u(\mu, t), \mu)$.
Assume that a reduced basis (RB) is available, $V := [V_1, \cdots, V_N]$, and $u \approx \hat{u} := Va$ with $a := a(t;\mu) \in \mathbb{R}^N$, then a reduced order model (ROM) can be obtained by using the Galerkin projection,

$$\hat{\Sigma} : \quad V^T F(\hat{u}) = V^T F(Va) = 0, \quad N \ll \mathcal{N}.$$

The corresponding output can be approximated by $\hat{y}(\mu) \approx y(\hat{u}(\mu))$.

Remark: Empirical Interpolation Method (EIM) or its variants (e.g. DEIM, empirical operator interpolation) can be employed if F is nonlinear/non-affine.

Review on RBM

POD-Greedy Algorithm

Algorithm 1 RB generation using POD-Greedy

Input: $\mathcal{P}_{train}, tol_{RB} (< 1)$

Output: RB: $V = [V_1, \ldots, V_N]$

Initialization: $\mathcal{W}^N = [\,]$, $N = 0$, $\mu_{\max} = \mu_0$, $\eta_N(\mu_{\max}) = 1$

while the error $\eta_N(\mu_{\max}) > tol_{RB}$ **do**

  Compute the trajectory $S_{\max} := \{u^n(\mu_{\max})\}_{n=0}^{K}$

  Enrich the RB: $\mathcal{W}^{N+1} := \mathcal{W}^N \oplus V_{N+1}$, where V_{N+1} is the first POD mode of the matrix $\bar{U} = [\bar{u}^0, \ldots, \bar{u}^K]$ with $\bar{u}^n := u^n(\mu_{\max}) - \Pi_{\mathcal{W}^N}[u^n(\mu_{\max})], n = 0, \ldots, K$, $\Pi_{\mathcal{W}^N}[u]$ is the projection of u on the current space $\mathcal{W}^N := \mathrm{span}\{V_1, \ldots, V_N\}$

  $N = N + 1$

  Find $\mu_{\max} := \arg\max_{\mu \in \mathcal{P}_{train}} \eta_N(\mu)$

end while

Influence factors of the cost: $\mathcal{N}$, $|\mathcal{P}_{train}|$, $\eta(\mu_{\max})$, K, ...

Ref: B. Haasdonk and M. Ohlberger, M2NA., 42, 2008.

Review on RBM

Empirical Interpolation (EI) & Collateral Reduced Basis (CRB)

Idea: $g(x,\mu) \approx \hat{g}_m := \sum_{i=1}^{m} \sigma_i(\mu) W_i, \quad \mu \in \mathcal{P}$ [Barrault et al. 2004]

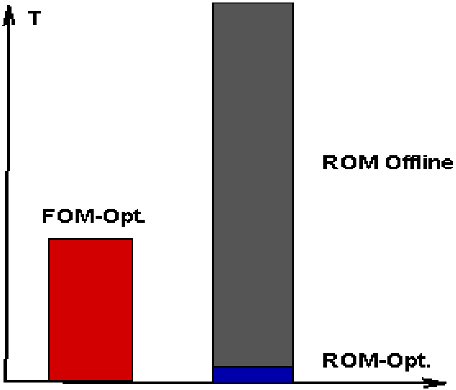
Algorithm 2 Generation of CRB and EI points T_M

Input: $L_{train}^{crb} := \{g(x,\mu) | \mu \in \mathcal{P}_{train}^{crb}\}, tol_{CRB} (< 1)$

Output: CRB: $W = [W_1, \ldots, W_M]$ and $T_M = \{x_1, \ldots, x_M\}$

Initialization: $m = 1, \mathcal{W}_{ei}^0 := \{0\}, \|r_0\| = 1$
while $\|\xi_{m-1}\| > tol_{CRB}$ **do**
  $\forall g \in L_{train}^{crb}$, define the 'best' approximation $\hat{g} = \sum_{i=1}^{m-1} \sigma_i W_i$ in the current space $\mathcal{W}_{ei}^{m-1} := \mathrm{span}\{W_1, \ldots, W_{m-1}\}$
  Define $\tilde{g} := \arg\max_{g \in L_{train}^{crb}} \|g - \hat{g}\|$, and the residual $\xi_m := \tilde{g} - \hat{\tilde{g}}$
  if $\|\xi_m\| \leq tol_{CRB}$ **then**
    Stop and set $M = m - 1$
  else
    Determine: $x_m := \arg\sup_{x \in \Omega} |r_m(x)|, \; W_m := \xi_m / \xi_m(x_m)$
  end if
  $m := m + 1$
end while

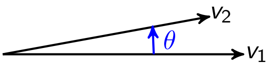
Review on RBM



Runtime comparison: FOM-Opt. vs. ROM-Opt.
33.88 h vs. 0.58(+82.25 = 82.83) h

Adaptive Snapshot Selection

The idea of ASS is to discard the redundant linear information in the trajectory earlier, before performing SVD for the generation of the basis.



Given non-zero vectors v_1, v_2,

$$\cos\theta = \frac{< v_1, v_2 >}{\|v_1\|\|v_2\|}.$$

- v_1 and v_2 are linearly dependent $\iff$ $|\cos\theta| = 1 \quad (\theta = 0 \text{ or } \pi)$

For a trajectory $\{u^n(\mu)\}_{n=0}^{n=K}$, define

$$Ind(u^n(\mu), u^m(\mu)) = 1 - \frac{|< u^n(\mu),\, u^m(\mu) >|}{\|u^n(\mu)\|\|u^m(\mu)\|},$$

ASS is implemented as follows.

ASS (cont.)

Algorithm 3 Adaptive snapshot selection (ASS)

Input: Initial vector $u^0(\mu)$, tol_{ASS}

Output: Selected snapshot matrix: $S^A = [u^{n_1}(\mu), u^{n_2}(\mu), \ldots, u^{n_\ell}(\mu)]$

Initialization: $j = 1$, $n_j = 0$, $S^A = [u^{n_j}(\mu)]$
for $n = 1, \ldots, K$ **do**
    Compute the vector $u^n(\mu)$.
    if $Ind(u^n(\mu), u^{n_j}(\mu)) > tol_{ASS}$ **then**
        $j = j + 1$
        $n_j = n$
        $S^A = [S^A, u^{n_j}(\mu)]$
    end if
end for

ASS (cont.)

Remark 1: It can be easily combined with other algorithms, e.g. Alg. 1, for the generation of RB and CRB.

Remark 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 more 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.

ASS-POD-Greedy

Algorithm 4 RB generation using ASS-POD-Greedy

Input: $\mathcal{P}_{train}, tol_{RB}$

Output: RB: $V = [V_1, \dots, V_N]$

1: Initialization: $\mathcal{W}^N = \{0\}$, $N = 0$, $\mu_{\max} = \mu_0$, $\eta(\mu_{\max}) = \infty$

2: **while** the error $\eta_N(\mu_{\max}) > tol_{RB}$ **do**

3: Compute the trajectory $S := \{u^n(\mu_{\max})\}_{n=0}^K$, adaptively select snapshots using Alg. 3, and get

$$S^A := \{u^{n_1}(\mu_{\max}), \dots, u^{n_\ell}(\mu_{\max})\}$$

4: Enrich the RB: $\mathcal{W}^{N+1} := \mathcal{W}^N \oplus V_{N+1}$, 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 $\bar{u}^{n_s} := u^{n_s}(\mu_{\max}) - \Pi_{\mathcal{W}^N}[u^{n_s}(\mu_{\max})], s = 1, \dots, n_\ell$ $(n_\ell \ll K)$, $\Pi_{\mathcal{W}^N}[u]$ is the projection of u on the current space $\mathcal{W}^N := \mathrm{span}\{V_1, \dots, V_N\}$

5: $N = N + 1$

6: Find $\mu_{\max} := \arg\max_{\mu \in P_{train}} \eta_N(\mu)$

7: **end while**

Remark: the error $\eta_N(\mu_{\max})$ can be an error estimator or the true error.

Error Estimation

Consider a general evolution scheme,

$$Au^{n+1}(\mu) = Bu^{n}(\mu) + g^{n},$$

where $A, B \in \mathbb{R}^{\mathcal{N}\times\mathcal{N}}$ are constant matrices,

$g^{n} := g(u^{n}(\mu), \mu) \in \mathbb{R}^{\mathcal{N}}$ is nonlinear or non-affine.

Let:

- $\hat{u}^{n}(\mu) = Va^{n}(\mu)$: RB approximation of $u^{n}(\mu)$,
- $\hat{g}^{n}(\mu) := \mathcal{I}_M[g(\hat{u}^{n}(\mu))] = W\beta^{n}(\mu)$: interpolation of g^{n},
- $V \in \mathbb{R}^{\mathcal{N}\times N}$, $W \in \mathbb{R}^{\mathcal{N}\times M}$: parameter-independent bases.
- $a^{n} \in \mathbb{R}^{N}$, $\beta^{n} \in \mathbb{R}^{M}$: parameter-dependent.

Define the residual:

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

We have the following propositions.

Error Estimation(cont.)

Proposition 1: Assume that the operator g is Lipschitz continuous, i.e., $\exists\ L_g \in \mathbb{R}^+$, s.t. $\|g(x) - g(y)\| \leq L_g \|x - y\|$, and the interpolation of g is 'exact' with a certain dimension of W, i.e.,

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

Assume the initial projection error $e^0 = 0$, then the field variable error $e^n := u^n - \hat{u}^n$ satisfies:

$$\|e^n(\mu)\| \leq \sum_{k=0}^{n-1} (\|A^{-1}\|)^{n-k} \mathtt{C}^{n-1-k} (\epsilon_{EI}^k(\mu) + \|r^{k+1}(\mu)\|),$$

where $\mathtt{C} = \|B\| + L_g$, and $\epsilon_{EI}^n(\mu)$ is the error due to the interpolation. A sharper error bound is given as:

$$\begin{aligned}\|e^n(\mu)\| &\leq \sum_{k=0}^{n-1} (\|A^{-1}B\| + L_g \|A^{-1}\|)^{n-1-k} (\|A^{-1}\| \epsilon_{EI}^k(\mu) + \|A^{-1} r^{k+1}(\mu)\|) \\ &:= \eta_{N,M}^n(\mu).\end{aligned}$$

Error Estimation(cont.)

Proposition 2: Under the assumptions of the Proposition 1, assume the output of interest $y(u^n(\mu))$ can be expressed as

$$y(u^n(\mu)) = Pu^n,$$

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:

$$\begin{aligned} \|e_O^{n+1}(\mu)\| \leq & \tilde{\eta}_{N,M}^{n+1} \\ & := (\|PA^{-1}B\| + L_g\|PA^{-1}\|)\eta_{N,M}^n \\ & \quad + \|PA^{-1}\|\epsilon_{EI}^n(\mu) + \|P\|\|A^{-1}r^{n+1}(\mu)\|. \end{aligned}$$

Remark: It is easy to show that the output error bound $\tilde{\eta}_{N,M}^{n+1}$ is sharper than the trivial bound

$$e_O^{n+1}(\mu) = P(u^{n+1} - \hat{u}^{n+1}) \leq \|P\|\|e^{n+1}(\mu)\| \leq \|P\|\|e^{n+1}(\mu)\|\eta_{N,M}^{n+1}.$$

RBM & ROM-based Optimization

FOM $\xrightarrow[\text{ASS-EI}]{\text{ASS-POD-Greedy}}$ **ROM**

$$\Sigma : \begin{cases} A c_z^{n+1} = B c_z^n + d_z^n \\ \qquad\qquad - \frac{1-\epsilon}{\epsilon} \Delta t h_z^n \\ q_z^{n+1} = q_z^n + \Delta t h_z^n \end{cases}$$

$$c_z^n, q_z^n \in \mathbb{R}^{\mathcal{N}}$$

Approx.

$$\hat{\Sigma} : \begin{cases} \hat{A}_{c_z} a_{c_z}^{n+1} = \hat{B}_{c_z} a_{c_z}^n + d_0^n \hat{d}_{c_z} \\ \qquad\qquad - \frac{1-\epsilon}{\epsilon} \Delta t \hat{H}_{c_z} \beta_z^n \\ a_{q_z}^{n+1} = a_{q_z}^n + \Delta t \hat{H}_{q_z} \beta_z^n \end{cases}$$

$$a_{c_z}^n, a_{q_z}^n \in \mathbb{R}^N, a_{q_z}^n \in \mathbb{R}^M$$

$$\mathcal{N} \gg N, M$$

RBM & ROM-based Optimization

FOM $\xrightarrow[\text{ASS-EI}]{\text{ASS-POD-Greedy}}$ **ROM**

$$\Sigma : \begin{cases} Ac_z^{n+1} = Bc_z^n + d_z^n \\ \qquad\qquad -\frac{1-\epsilon}{\epsilon}\Delta t h_z^n \\ q_z^{n+1} = q_z^n + \Delta t h_z^n \end{cases} \quad c_z^n, q_z^n \in \mathbb{R}^{\mathcal{N}}$$

Approx.

$$\hat{\Sigma} : \begin{cases} \hat{A}_{c_z} a_{c_z}^{n+1} = \hat{B}_{c_z} a_{c_z}^n + d_0^n \hat{d}_{c_z} \\ \qquad\qquad -\frac{1-\epsilon}{\epsilon}\Delta t \hat{H}_{c_z}\beta_z^n \\ a_{q_z}^{n+1} = a_{q_z}^n + \Delta t \hat{H}_{q_z}\beta_z^n \end{cases} \quad a_{c_z}^n, a_{q_z}^n \in \mathbb{R}^N, a_{q_z}^n \in \mathbb{R}^M$$

$\mathcal{N} \gg N, M$

$$\min_{\mu \in \mathcal{P}} \{-Pr(c_z(\mu), q_z(\mu); \mu)\} \; s.t.$$
$$Rec(c_z(\mu), q_z(\mu); \mu) \geq Rec_{min},$$
$$c_z(\mu), q_z(\mu) : \text{ solutions to } \Sigma.$$

Approx.

$$\min_{\mu \in \mathcal{P}} \{-\hat{Pr}(\hat{c}_z(\mu), \hat{q}_z(\mu); \mu)\} \; s.t.$$
$$\hat{Rec}(\hat{c}_z(\mu), \hat{q}_z(\mu); \mu) \geq Rec_{min},$$
$$\hat{c}_z(\mu), \hat{q}_z(\mu) : \text{ solutions to } \hat{\Sigma}.$$

Num. Ex.: Performance of ASS

Generation of CRBs (W_a, W_b) with different tol_{ASS} at the same tolerance $tol_{CRB} = 1.0 \times 10^{-7}$.

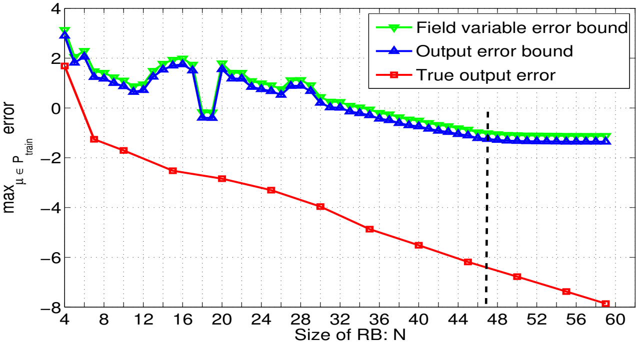
	tol_{ASS}	M (W_a W_b)		Runtime [h]
no ASS	–	146	152	62.5 (-)
ASS	1.0×10^{-4}	147	152	6.05 (-90.3%)
ASS	5.0×10^{-4}	145	148	4.06 (-93.5%)
ASS	1.0×10^{-3}	147	151	3.52 (-94.4%)

Runtime comparison of using POD-Greedy algorithm with early-stop (Alg. 5) with and without ASS.

Simulations	Num. of RB (N)	Runtime [h]
POD-Greedy	45	19.75 [1]
ASS-POD-Greedy	47	9.86 (-50.1%)

[1] Done on a Workstation with 4 Intel Xeon E7-8837 CPUs (8 core per CPU) 2.67 GHz RAM 1TB, others were done on the PC with Quad CPU 2.83GHz RAM 4GB.

Num. Ex.: Performance of Error Estimation



Comparison of the error bound decay during the RB extension and the corresponding true output error.

Output error bound: $\tilde{\eta}_N := \max_{z\in\{\mathrm{a,b}\}}\{\max_{\mu\in\mathcal{P}_{train}} \bar{\tilde{\eta}}^K_{N,M,c_z}(\mu)\}$
Field variable error bound: $\eta_N := \max_{z\in\{\mathrm{a,b}\}}\{\max_{\mu\in\mathcal{P}_{train}} \bar{\eta}^K_{N,M,c_z}(\mu)\}$
True output error: $e_N^{\max} := \max_{\mu\in\mathcal{P}_{train}} \bar{e}_N(\mu)$, where
$\bar{e}_N(\mu) := \max_{z\in\{\mathrm{a,b}\}} \bar{e}_{N,c_z}(\mu)$, $\bar{e}_{N,c_z}(\mu) := \frac{1}{K}\sum_{n=1}^{K} \|c^n_{z,O}(\mu) - \hat{c}^n_{z,O}(\mu)\|$

Num. Ex.: ASS-POD-Greedy with Early-stop

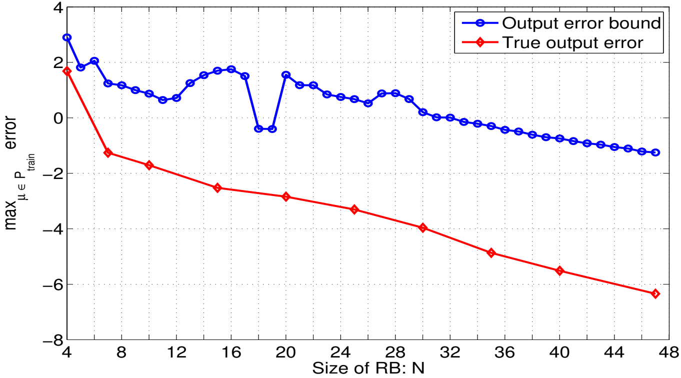
Algorithm 5 RB generation using ASS-POD-Greedy with early-stop

Input: $\mathcal{P}_{train}, tol_{RB}, tol_{decay}$

Output: RB: $V = [V_1, \dots, V_N]$

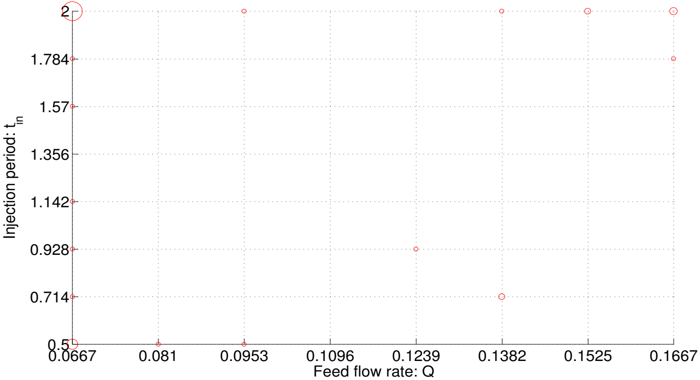
Implement Step 1 in Alg. 4
while the error $\eta_N(\mu_{\max}) > tol_{RB}$ **do**
    Implement Steps 3-6 in Alg. 4
    Compute the decay of the error bound $d\eta = \frac{\eta_{N-1}(\mu_{\max}^{old}) - \eta_N(\mu_{\max})}{\eta_{N-1}(\mu_{\max}^{old})}$
    if $d\eta < tol_{deacy}$ **then**
        Compute the true output error at the selected parameter $\mu_{\max}$, $\bar{e}_N(\mu_{\max})$
        if $\bar{e}_N(\mu_{\max}) < tol_{RB}$ **then**
            Stop
        end if
    end if
end while

Num. Ex.: Error Decay



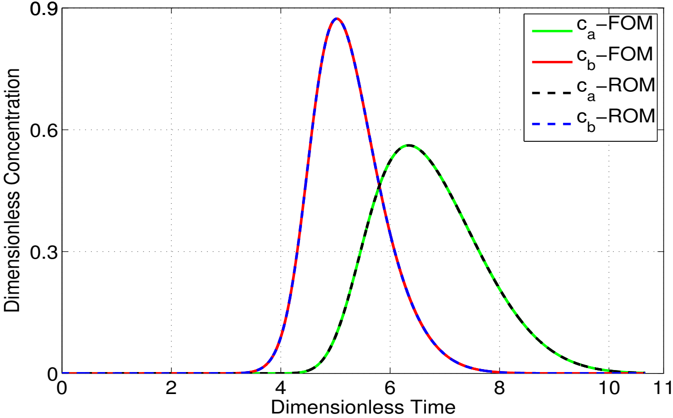
Error bound decay during the RB extension using Alg. 5 and the corresponding maximal true output error.

Num. Ex.: Parameter Location



Parameter location during the RB extension using Alg. 5.

Num. Ex.: ROM Performance



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

Num. Ex.: ROM Validation

Runtime comparison of the detailed and reduced simulation over a validation set P_{val} with 600 random sample points. Tolerance for the generation of ROM: $tol_{CRB} = 1 \times 10^{-7}, tol_{RB} = 1 \times 10^{-6}, tol_{ASS} = 1 \times 10^{-4}$.

Simulations	Max. error	Average runtime [s]/FoS
FOM ($\mathcal{N} = 1500$)	–	312.13(-)
ROM, no ASS for POD-Greedy	3.46×10^{-7}	5.85 / 53
ROM, ASS-POD-Greedy	4.11×10^{-7}	6.43 / 48

Num. Ex.: ROM-based Optimization

The optimization for batch chromatography:

FOM-based Opt.:

$$\min_{\mu\in\mathcal{P}} \{-Pr(c_z(\mu), q_z(\mu); \mu)\} \ s.t.$$
$$Rec(c_z(\mu), q_z(\mu); \mu) \geq Rec_{min},$$
$$c_z(\mu), q_z(\mu) : \text{ solutions to } \Sigma.$$

$\xleftarrow{\text{Approx.}}$

ROM-based Opt.:

$$\min_{\mu\in\mathcal{P}} \{-\hat{P}r(\hat{c}_z(\mu), \hat{q}_z(\mu); \mu)\} \ s.t.$$
$$\hat{Rec}(\hat{c}_z(\mu), \hat{q}_z(\mu); \mu) \geq Rec_{min},$$
$$\hat{c}_z(\mu), \hat{q}_z(\mu) : \text{ solutions to } \hat{\Sigma}.$$

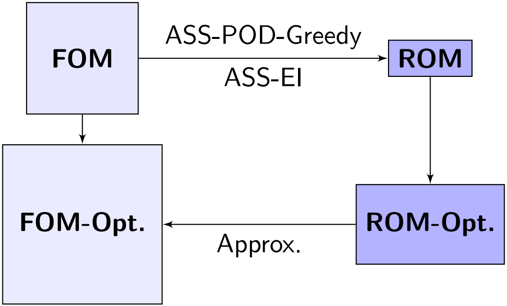
- $\mathcal{P} = [0.0667,\ 0.1667] \times [0.5,\ 2.0]$
- $Pu_{\mathrm{a}} = Pu_{\mathrm{b}} = 95.0\%, Rec_{min} = 80.0\%$
- $\mathcal{N} = 1500, N = 47$

Comparison of the optimization based on the ROM and FOM.

Simulations	Obj. (Pr)	Opt. solution (μ)	It.	Runtime [h]/FoS
FOM-Opt.	0.020264	$(0.07964, 1.05445)$	202	33.88 / -
ROM-Opt.	0.020266	$(0.07964, 1.05445)$	202	0.64 / 53

$\star$ The optimizer: NLOPT_GN_DIRECT_L

Conclusions and Outlook

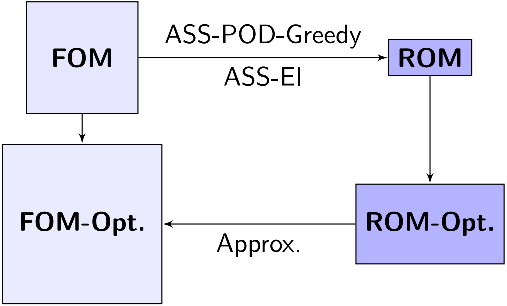


- Adaptive Snapshots Selection
- Output-oriented error estimation
- Early-stop for (ASS-)POD-Greedy algorithm

Outlooks:

- Error estimation with dual system
- RBM to Simulated Moving Bed (SMB) chromatography
- RBM for SMB with uncertainty quantification (UQ)

Conclusions and Outlook



- Adaptive Snapshots Selection
- Output-oriented error estimation
- Early-stop for (ASS-)POD-Greedy algorithm

Outlooks:

- Error estimation with dual system
- RBM to Simulated Moving Bed (SMB) chromatography
- RBM for SMB with uncertainty quantification (UQ)

Thank you for your attention!