Be greedy and learn: efficient and certified algorithms for parametrized optimal control problems

Martin Lazar
University of Dubrovnik

The Mathematics of Scientific Machine Learning and Digital Twins
Erice, November 2025

Joint work with:
Hendrik Kleikamp (Graz), Cesare Molinari (Genova)

Parameter dependent problems

Linear systems (of finite dimension N) associated to a parameter $\nu \in \mathcal{K}$

$$A_\nu x = b_\nu,$$

- $A_\nu : X \to Y$
- X, Y Banach spaces.

We suppose the equations are well posed for each ν.
x_ν – corresponding solution.

The goal

To tackle the problem in an efficient and robust manner:

- model reduction (greedy algorithms)
- machine learning (DNN, etc)

The greedy method[1]

X – a Banach space
$K \subset X$ – a compact subset.
The method approximates K by a a series of finite dimensional linear spaces V_k (a linear method).

Goal of the greedy algorithm

Find a reduced space $V_k \subset X$ of (small) dimension k such that

$$\mathrm{dist}(V_k, K) \leq \varepsilon,$$

where $\varepsilon > 0$ is the approximation tolerance.

Two phases:

- selection of reduced basis vectors x_i (offline)
- approximation of an arbitrary $x \in K$ by $\sum \alpha_i x_i \in V_k$ (online).

[1] A. Cohen, R. DeVore, A. T. Patera, etc,

The algorithm - offline part

Fix finite *training set* $K_{tr} \subset K$.

The first step

Choose $x_1 \in K_{tr}$ such that

$$\|x_1\|_X = \max_{x \in K_{tr}} \|x\|_X .$$

The general step

Having found $x_1..x_n$, denote $V_n = \operatorname{span}\{x_1, \ldots, x_n\}$.
Choose the next element

$$x_{n+1} := \arg \max_{x \in K_{tr}} \operatorname{dist}(x, V_n) .$$

The algorithm stops

when $\sigma_n(K) := \max_{x \in K_{tr}} \operatorname{dist}(x, V_n)$ becomes less than the given tolerance ε.

Important:

In each iteration elements $x \in K_{tr}$ are "projected" to V_n (e.g. by Galerkin)

$$x \approx x^* = \sum_1^n \alpha_i(x)\, x_i .$$

α_i - projection coefficients.

Efficiency

In order to estimate **the efficiency of the greedy algorithm** we compare its approximation rates $\sigma_n(K)$ with the best possible one.

The Kolmogorov n width, $d_n(K)$

– measures how well K can be approximated by a subspace in X of a fixed dimension n.

$$d_n(K) := \inf_{\dim Y = n} \sup_{x \in K} \inf_{y \in Y} \|x - y\|_X \, .$$

Thus $d_n(K)$ represents optimal approximation performance that can be obtained by a n-dimensional linear space.

Theorem[1]
The greedy approximation rates have same decay as the Kolmogorov widths.

For any $\alpha > 0, C_0 > 0$

$$d_n(K) \leq C_0 n^{-\alpha} \quad \Longrightarrow \quad \sigma_n(K) \leq C_1 n^{-\alpha}, \quad k \in \mathbf{N},$$

where $C_1 := C_1(\alpha, C_0)$.

[1] A. Cohen, R. DeVore, Approximation of high-dimensional parametric PDEs, *Acta Numerica*, **24** (2015) 1–159.

Greedy approach – implementation issues

Let us go back to the problem of parameter dependent linear equations

$$A_\nu x = b_\nu, \tag{1}$$

- $A_\nu : X \to Y$
- X, Y Banach spaces.

The solution manifold K

$K := \{x_\nu : \nu \in \mathcal{K},\ x_\nu \text{ solves } (1)\} \subset X$

Let us approximate K by the greedy procedure.
Suppose

- we have chosen the training set $K_{tr} \subseteq K$ determined by the finite set of parameters $\{\nu_1, \dots, \nu_N\}$.
- Suppose we have chosen ν_1 and calculate x_1.

How should we estimate

$$\mathrm{dist}_{x_\nu \in K_{tr}}(x_\nu, [x_1])$$

without knowing the solution x_ν?

Take the residual of the linear system as a distance estimator:

$$r_\nu(x) := \|A_\nu x - b_\nu\|_Y, \quad x \in X$$

Theorem (Distance estimator for an unknown solution)

If both A_ν and A_ν^{-1} are uniformly bounded

$$c\,\|x_\nu - x\|_X \qquad \leq r_\nu(x) \quad \leq \quad C\,\|x_\nu - x\|_X,$$

with

- $C = \sup_\nu \|A_\nu\|$
- $c^{-1} = \sup_\nu \|A_\nu^{-1}\|$

- Instead of maximising the (unknown) distances to the approximation subspace, we maximise the residuals.
- The procedure fulfils the requirements of the greedy theory and stops after the most n (system dimension) steps.

Online computations of the greedy reduced order model (G ROM)

For a new parameter ν we approximate the (unknown) solution to $A_\nu x = b_\nu$ by:

- $\tilde{x}_\nu^k = \sum_{i=1}^k \alpha_i^\nu x_i \in V_k$ such that

$$\sum_{i=1}^{k} \alpha_i^\nu \underbrace{A_\nu x_i}_{z_i^\nu} \approx b_\nu.$$

This requires

- computing $z_i^\nu = A_\nu x_i$ for $i = 1, \ldots, k$
- projecting b_ν onto $\bar{Z}_\nu = [z_1^\nu, \cdots, z_k^\nu]$

- Costly part in the online phase (for large systems) :
  Computation of $\quad A_\nu x_i \quad$ for $i = 1, \ldots, k$.
- Instead: Learn the **parameter to coefficients map** π

$$\nu \quad \mapsto \quad \pi_k(\nu) \coloneqq (\alpha_i^\nu)_{i=1}^k$$

by machine learning surrogate $\pi_k^* \colon \mathcal{K} \to \mathbb{R}^k$.

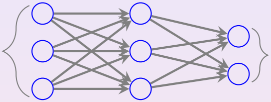
N.B.

- The coefficients $(\alpha_i^\nu)_{i=1}^k$ have already been calculated for all the training parameters in the offline phase.
- Training data is automatically generated during the greedy algorithm.
- Machine learning surrogate is trained during the offline phase.

Machine learning approaches

Various machine learning methods can be applied here, we considered:

- Deep neural networks (DNN), see for instance [e.g. Petersen, Voigtlaender'18].

ν  $\pi_k^*(\nu) \in \mathbf{R}^k$

- Kernel methods (VKOGA), see for instance [e.g.Santin, Haasdonk'21].

$$\pi_k^*(\nu) = \sum_{i \in \Xi} \alpha_i k(\nu, x_i)$$

- Gaussian process regression (GPR), see for instance [Rasmussen, Williams'06].

$$\pi_k^*(\nu) = \mathbb{E}_y[P(y|\nu, X, Y)]$$

Parametrized optimal control problems

Control systems (of finite dimension n) associated to a parameter $\nu \in \mathcal{K}$

$$\dot{x}_\nu(t) = \mathcal{A}_\nu x_\nu(t) + \mathcal{B}_\nu u_\nu(t) \text{ for } t \in [0,T], \quad x_\nu(0) = x_\nu^0.$$

accompanied by optimization problems

$$\min_u \mathcal{J}_\nu(u) := \frac{1}{2}\left[\underbrace{\langle M\left(x_\nu(T) - x_\nu^T\right), x_\nu(T) - x_\nu^T\rangle}_{\text{deviation from the target state } x_\nu^T} + \underbrace{\int_0^T \langle Ru(t), u(t)\rangle dt}_{\text{energy of the control}}\right]$$

with $M \geq 0, R \geq \alpha I$ for some $\alpha > 0$

The solution is determined by the optimal adjoint vector $\varphi_\nu(T)$ satisfying

Linear system

$$\underbrace{\left(I + M\Lambda_\nu^R\right)}_{A_\nu} \underbrace{\varphi_\nu(T)}_{x_\nu} = \underbrace{x_\nu^T - e^{\mathcal{A}_\nu T} x_\nu^0}_{b_\nu},$$

where $\Lambda_\nu^R \in \mathcal{L}(X, X)$ is the weighted controllability Gramian.

Approximating the solution manifold by linear subspaces

- Manifold $\mathcal{M} := \{\varphi_\nu(T) : \nu \in \mathcal{K}\} \subset X = \mathbf{R}^N$
- Approximation tolerance $\varepsilon > 0$

Goal of the greedy algorithm

Find a reduced space $V_k \subset X$ of (small) dimension k such that

$$\mathrm{dist}(V_k, \mathcal{M}) \leq \varepsilon.$$

- Apply a greedy algorithm to construct a reduced basis for the optimal final time adjoint states (by means of an efficient error estimator for the reduced space).
- Later: Accelerate online phase using machine learning with error certification.

[2] H. Kleikamp, M.L, C. Molinari, Be greedy and learn: efficient and certified algorithms for parametrized optimal control problems, ESAIM: M2AN, (2025)

We use ML to approximate the greedy approximation of the solution.

Error estimates

- A priori bound:

$$\|\tilde{\varphi}_\nu^k - \varphi_\nu(T)\| \quad \leq \quad C_\Lambda \underbrace{\varepsilon}_{\substack{\text{greedy}\\ \text{tolerance}}} + \underbrace{\|\pi_k(\nu) - \pi_k^*(\nu)\|}_{\substack{\text{approximation error}\\ \text{of machine learning}}}.$$

  with $C_\Lambda = \sup_\nu \|I + \Lambda_\nu^R\|_{\mathcal{L}(X,X)}$.

- A posteriori bound:

$$\|\tilde{\varphi}_\nu^k - \varphi_\nu(T)\| \quad \leq \quad r_\nu(\tilde{\varphi}_\nu^k) \quad \leq \quad \|I + \Lambda_\nu\| \|\tilde{\varphi}_\nu^k - \varphi_\nu(T)\|.$$

- Problem definition:

$$\begin{aligned} \partial_t v_\nu(t,y) - \nu_1 \Delta v_\nu(t,y) &= 0 && \text{for } t \in [0,T], y \in \Omega, \\ v_\nu(t,0) &= u_{\nu,1}(t) && \text{for } t \in [0,T], \\ v_\nu(t,1) &= u_{\nu,2}(t) && \text{for } t \in [0,T], \\ v_\nu(0,y) = v_\nu^0(y) &= \sin(\pi y) && \text{for } y \in \Omega. \end{aligned}$$

- Semi-discretisation, with $n = 100$.
- Weighting matrices:

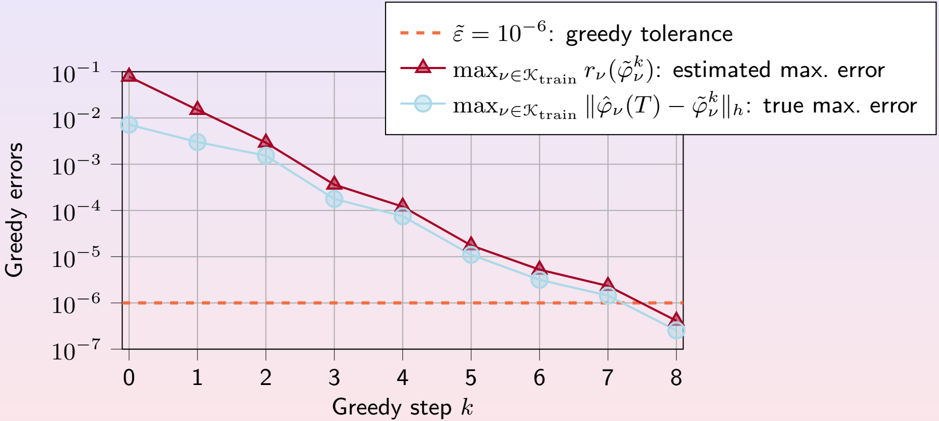
$$M = I \in \mathbb{R}^{n\times n} \qquad \text{and} \qquad R = \begin{bmatrix} 0.125 & 0 \\ 0 & 0.25 \end{bmatrix}$$

- Target state:

$$v_\nu^T(y) = \nu_2 y$$

- $\Omega = [0,1]$, $T = 0.1$,
- Parameter set $\mathcal{K} = [1,2] \times [0.5, 1.5]$

Results of running the greedy algorithm with 64 uniformly distributed training parameters:



Numerical example: Results

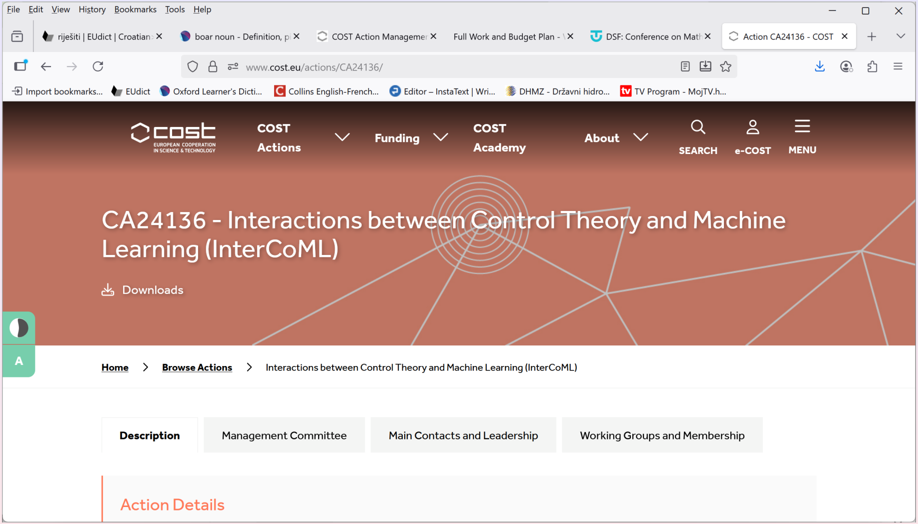
Results on a set of 100 randomly drawn test parameters:

Method	Avg adjoint error	Averaged control error	Avg. runtime (s)	Avg. speedup
Exact solution	–	–	6.2760	–
G ROM	$5.3 \cdot 10^{-8}$	$5.4 \cdot 10^{-9}$	2.6526	2.37
DNN ROM	$5.8 \cdot 10^{-6}$	$2.0 \cdot 10^{-6}$	0.1623	40.33
VKOGA ROM	$1.8 \cdot 10^{-5}$	$6.9 \cdot 10^{-6}$	0.1580	41.03
GPR ROM	$2.2 \cdot 10^{-6}$	$7.6 \cdot 10^{-7}$	0.1572	41.40

Conclusion:

- the novel approach that combines the standard ROM with ML tools
- significant speed up of the online phase
- error bounds available.

Announcement - new COST Action



Thanks for your attention!