Received: xxx **Accepted:** xxx **Published:** xxx.
DOI. xxxxxxx
Volume xxx, Issue xxx, p.p. 1-23: xxx
Research Article

Open Access

Control and Optimization in Applied Mathematics - COAM

Learning-Assisted Adaptive Krylov Solvers for Large-Scale Matrix Differential Riccati Equations

Yaprak Güldoğan Dericioğlu

Bitlis Eren University, Department of Mathematics, Bitlis, Türkiye

Correspondence:
Yaprak Güldoğan Dericioğlu
E-mail:
yguldogan@beu.edu.tr

Abstract. The Extended Block Arnoldi–Backward Differentiation Formula (EBA–BDF) is a projection-based integrator for solving large-scale Matrix Differential Riccati Equations (DREs). Like many Krylov subspace methods, its performance depends on the choice of the subspace dimension. In practice, this parameter is often determined through empirical tuning. In this work, we introduce a lightweight, data-driven pre-solver to estimate this dimension *a priori*. The approach uses a Random Forest model trained on spectral norms and discretization parameters, and predicts the required subspace size without modifying the numerical core or stability properties of the original method. Numerical experiments show that the proposed approach can automate parameter selection and reduce the need for manual tuning. The effect is more noticeable in diffusion-dominated regimes, where spectral properties lead to more regular Krylov convergence. By simplifying the initialization stage, the approach supports the practical use of EBA–BDF solvers in large-scale problems.

How to Cite
Güldoğan Dericioğlu, Y. (2027). "Learning-assisted adaptive Krylov solvers for large-scale matrix differential riccati equations". *Control and Optimization in Applied Mathematics*, 12(-), 1-23. https://doi.org/10.30473/coam.2026.76340.1355

Keywords. Low-rank differential Riccati equation, Krylov subspace projection, Backward differentiation formula, Learning-assisted solvers, Learning-assisted numerical solvers, Spectral descriptors, Machine learning.

MSC. 65F30; 49N10; 65L05.

1 Introduction

The differential Riccati equation (DRE) is a nonlinear matrix differential equation that appears in many areas of control and estimation theory [1, 17]. Such problems become particularly challenging in large-scale or ill-conditioned settings arising in modern control applications [8]. A general continuous-time DRE can be expressed as

$$\begin{cases} \dot{X}(t) &= A^T X(t) + X(t) A - X(t) B R^{-1} B^T X(t) + Q, \\ X(t_0) &= X_0, \qquad t \in [t_0, t_f], \end{cases} \tag{1}$$

where $A \in \mathbb{R}^{n\times n}$ is large, sparse, and nonsingular; $B \in \mathbb{R}^{n\times s}$ has full rank with $s \ll n$; $R \in \mathbb{R}^{s\times s}$ is symmetric positive definite; and $Q \in \mathbb{R}^{n\times n}$ is symmetric positive semidefinite. The initial condition X_0 is a given low-rank symmetric matrix.

Several numerical schemes have been developed for the efficient integration of (1). Early efforts employed implicit Runge–Kutta methods—known to preserve key structure and behave well for stiff problems in this context [11]—and later widely used BDF discretizations for DREs [2, 3]. Although these methods are accurate, the large matrix operations required at each step make them impractical for high-dimensional systems. To overcome this difficulty, projection-based and low-rank techniques were introduced, taking advantage of the fact that the DRE solution often has low numerical rank. Representative examples include the low-rank Alternating Direction Implicit (LR–ADI) method and the family of Krylov subspace projection methods [4, 5, 13, 21], which achieve high efficiency for both algebraic and differential Riccati equations. A recent review on numerical methods for large-scale low-rank differential matrix equations is provided in [15]. More recently, learning-based approaches have been proposed to enhance iterative solvers and preconditioning strategies, including neural preconditioners and data-driven Krylov methods [9, 10, 18].

Within this context, the Extended Block Arnoldi–Backward Differentiation Formula (EBA–BDF) method [14] provides a projection-based integrator that combines an extended Krylov subspace with a BDF discretization to obtain low-rank approximations. Its numerical stability and efficiency are well established. However, the choice of the Krylov subspace dimension m remains a practical parameter that directly controls the balance between computational cost and approximation accuracy.

In typical implementations, m is selected through empirical tuning or repeated solver runs. This trial-and-error process becomes increasingly expensive for large-scale problems, where each additional Arnoldi step contributes to both memory usage and orthogonalization cost. Automating this choice is therefore not only a matter of convenience, but also a way to reduce unnecessary computational overhead.

Previous studies on adaptivity in Krylov solvers have mainly focused on *a posteriori* mechanisms that adjust the subspace dimension during the iteration. Examples include adaptive

rational Krylov methods [13] and residual-based restart strategies that dynamically expand or restart the subspace. Related approaches include learning-based techniques for improving iterative solvers, such as neural preconditioning methods [16]. In contrast, estimating a suitable subspace dimension before integration begins provides a complementary perspective. An accurate *a priori* estimate can reduce over-iteration and improve solver initialization, particularly in settings where repeated simulations are required.

To address this limitation, we introduce a lightweight data-driven pre-solver that predicts the Krylov subspace dimension prior to integration. The predictive component operates independently of the numerical solver and does not modify the EBA–BDF algorithm itself. Instead, it provides an informed initial value for m based on operator descriptors, allowing the projection process to start closer to an efficient regime.

The framework focuses on estimating the Krylov subspace dimension before the integration stage, rather than adjusting it during the iteration. This perspective complements existing adaptive Krylov methods by shifting part of the computational effort to an offline prediction step. In this way, unnecessary subspace expansions can be avoided when a suitable initial dimension is available.

To the best of our knowledge, the a priori prediction of the Krylov subspace dimension has not been studied specifically in the context of the EBA–BDF low-rank integrator. Existing approaches either adapt the subspace dimension during the iteration or attempt to approximate the solution itself using data-driven models. Here, the learning component is restricted to predicting the subspace size, while the numerical solver remains unchanged.

The training data are generated from EBA–BDF(2) simulations covering different matrix sizes, diffusion coefficients, and conditioning levels. Each instance provides descriptors such as matrix dimension, Frobenius norm, condition number, spectral radius, and solver tolerance. These quantities are used as input features for a Random Forest regressor [7], which learns the relationship between operator properties and the required subspace dimension.

The results show that the predictor provides accurate initialization in diffusion-dominated settings, where Krylov convergence tends to be more regular, while reduced accuracy is observed in convection–reaction regimes with stronger spectral irregularities. This observation aligns with the theoretical arguments presented in Section 2.1, particularly regarding the dependence of Krylov convergence on spectral structure.

The main contributions are summarized as follows:

- We introduce a learning-assisted framework for estimating the Krylov subspace dimension in EBA–BDF solvers before integration.
- A dataset of differential Riccati problems is constructed from multiple operator families with varying spectral characteristics.

- A Random Forest regression model is trained on physically interpretable features and evaluated across different regimes.
- The validity domain of the approach is analyzed, showing reduced accuracy for spectrally irregular operators.
- The predictive component is solver-agnostic and does not alter the numerical core of the EBA–BDF method.

This approach provides a practical way to initialize projection-based solvers for large-scale differential matrix equations.

The remainder of the paper is organized as follows. Section 2 reviews the EBA–BDF algorithm and the data-collection process and presents the regression design and feature selection strategy. Section 3 reports the numerical experiments and evaluates the model under various spectral regimes. Section 4 discusses strengths and limitations, while Section 5 concludes the paper and outlines future work.

2 Methodology

2.1 Background on the EBA–BDF Method

The Extended Block Arnoldi–Backward Differentiation Formula (EBA–BDF) method, introduced in [14], combines projection onto an extended block Krylov subspace with an implicit BDF time integrator. For large-scale differential Riccati equations, where the full matrix solution cannot be formed or stored, this projection structure is what makes the method computationally viable. Throughout this work, we employ the second-order variant, referred to as EBA–BDF(2), while the term EBA–BDF is used in the general context without reference to a specific order.

Consider the differential Riccati equation

$$\dot{X} = A^T X + X A - X B R^{-1} B^T X + Q, \qquad X(t_0) = X_0.$$

To enable a low-rank formulation, we assume that Q admits a factorization $Q = C^T C$, where $C \in \mathbb{R}^{s\times N}$, $s \ll N$, has a small number of rows. In the numerical experiments, we set $R = I$ for simplicity, and represent the initial condition in factorized form as $X_0 = Z_0 Z_0^T$. Rather than solving this equation directly in $\mathbb{R}^{N\times N}$, the EBA–BDF method seeks an approximation in the extended Krylov subspace

$$\mathcal{K}_m^e(A, C^T) = \operatorname{span}\{C^T, A C^T, \ldots, A^{m-1} C^T, A^{-1} C^T, \ldots, A^{-(m-1)} C^T\}.$$

After m Arnoldi steps, let V_m denote the orthonormal basis of this subspace and define $T_m = V_m^T A V_m$. The approximate solution takes the form

$$X_m(t) = V_m Y_m(t) V_m^T,$$

where $Y_m(t)$ solves the reduced equation

$$\dot{Y}_m = T_m Y_m + Y_m T_m^T - Y_m B_m B_m^T Y_m + C_m^T C_m,$$

with $B_m = V_m^T B$ and $C_m = C V_m$.

From a computational standpoint, the primary bottleneck is not time integration itself, but rather the construction and orthogonalization of the Krylov basis. The subspace dimension m dictates this cost: a value that is too small results in poor approximation accuracy, whereas an excessively large value inflates both memory requirements and increased orthogonalization overhead. In large-scale applications, even a slight overestimation of m can significantly escalate runtime — a challenge that motivates the present study.

In typical implementations, m is selected through trial-and-error or adaptive enlargement, each requiring repeated solver executions. We propose that this overhead can be bypassed: by obtaining a reliable estimate of m prior to integration, the solver can be initialized directly at an efficient subspace dimension. To achieve this, we introduce a data-driven pre-solver that predicts m based on inexpensive algebraic and spectral descriptors of A such as its norm, condition number, and spectral radius.

This predictive layer does not alter the EBA–BDF algorithm itself. All stability and convergence properties of the original method are preserved; the predictor acts solely as an initialization mechanism. The same idea extends naturally to other projection-based integrators, since dependence on the Krylov dimension is a structural feature shared across this class of methods.

Theoretical rationale. The rationale for learning m from spectral descriptors rests on three known properties of Krylov projection methods, which we recall from the literature to motivate the feature design in Section 2.3.

1. **Residual–error relation.** Let $X_m(t) = V_m Y_m(t) V_m^T$ denote the EBA–BDF approximation and $E_m(t) = X(t) - X_m(t)$ represent the approximation error. According to Theorem 5.3 in [14], the error E_m satisfies the following differential equation:

$$\dot{E}_m = (A^T - XBB^T)E_m + E_m(A - BB^T X) + E_m BB^T E_m + R_m,$$

where R_m is the projection residual. For small $\|E_m\|$, the quadratic term becomes negligible, implying that the evolution of E_m is driven primarily by R_m. Consequently, controlling $\|R_m(T_f)\|_F$ serves as a reliable surrogate for the approximation error, justifying the stopping criterion $\|R_m(T_f)\|_F \leq$ tol employed in Algorithm 1.

2. **Monotonicity with respect to m.** Because the extended Krylov spaces are nested, $\mathcal{K}_m^e \subset \mathcal{K}_{m+1}^e$, increasing m cannot degrade the approximation quality [12, 21]:

$$\|R_{m+1}(T_f)\|_F \leq \|R_m(T_f)\|_F.$$

However, the *rate* of improvement varies considerably across problem regimes, which is precisely what makes m difficult to predict from a single descriptor.

3. **Regime-dependent convergence.** When the spectrum of A is well clustered along the negative real axis—as in diffusion-dominated problems—the residual exhibits near-geometric decay with respect to m [12]. For convection–reaction dominated systems, nonnormal effects produce less regular convergence, and matrices with similar global descriptors may require markedly different Krylov dimensions [21]. This observation is the key motivation for our approach: since m cannot be inferred from a single scalar such as $\|A\|_F$ or $\rho(A)$, a richer feature representation capturing spectral clustering, conditioning, and nonnormality jointly is needed.

Note that if $m_{\text{pred}} \geq m_{\text{true}}$, the stopping criterion is met without restarts; the predictor should therefore be understood as a tool for reducing adaptive corrections, not as a replacement for the underlying solver.

Remark. The descriptors used here characterize A alone. In principle, the Riccati dynamics — particularly the quadratic term — can also influence the effective spectral properties governing convergence. Incorporating such problem-specific information represents a promising direction for improving predictive accuracy in nonnormal and strongly nonlinear regimes.

2.2 Data Generation and System Characterization

The dataset is constructed by running the EBA–BDF(2) solver on a collection of differential Riccati systems under varying configurations. For each run, the minimal number of Arnoldi iterations m required to satisfy the stopping criterion is recorded as the target variable. In the numerical setting, n_0 denotes the number of grid points along each spatial dimension and $N = n_0^2$ the total system dimension, so that $A \in \mathbb{R}^{N\times N}$; this corresponds to the variable n used in the general formulation of Section 1.

Rather than working with random matrices, we deliberately focus on two physically motivated operator classes: *(i)* convection–diffusion–reaction systems and *(ii)* diffusion-dominated systems. This choice is driven by the fact that these two classes exhibit fundamentally different spectral structures — well-clustered spectra in the diffusion-dominated case versus nonnormal,

irregularly distributed spectra in the convection–reaction case — which in turn produce qualitatively different Krylov convergence behavior. Covering both regimes ensures that the training data reflects the full range of difficulty the predictor is expected to handle.

Within each class, spatially varying coefficients are defined using constant, oscillatory, or linear profiles, and the resulting operators are discretized by second-order finite differences. Low-rank factors $B \in \mathbb{R}^{N\times s}$ and $C \in \mathbb{R}^{s\times N}$, $s \ll N$, are generated with fixed random seeds to ensure reproducibility.

For each configuration, the following descriptors are recorded: grid resolution n_0, time step h, tolerance tol, Frobenius norm $\|A\|_F$, condition number $\kappa(A)$, spectral radius $\rho(A)$, and the resulting Krylov dimension m. The variable `PDE_ID` is a categorical indicator encoding the operator class (diffusion-dominated or convection–reaction). The spectral radius and condition number directly govern convergence rate, while $\|A\|_F$ captures overall operator scaling — together they provide the information that classical Krylov theory identifies as most relevant to subspace size selection.

The dataset comprises approximately one hundred instances. Although modest in size, each sample corresponds to a complete large-scale Riccati solve; the dataset is therefore better understood as a structured collection of numerical experiments than as a conventional statistical sample. This distinction matters for how generalization should be interpreted: the model is not interpolating between random draws, but extrapolating across physically distinct problem regimes.

The train–test split reflects this structure. Rather than a random partition, we use disjoint parameter configurations: the test set contains combinations of (h, tol) and grid sizes n_0 that are entirely absent from training. This design deliberately forces the model to generalize across discretization regimes, providing a more demanding and meaningful evaluation than interpolation-based splits would allow.

Data sampling. Two representative subsets are used in the experiments:

1. **Diffusion-dominated systems:**

$$n_0 \in \{30,\ 40, \ldots, 100\}, h \in \{5\times10^{-3}, 2\times10^{-3}, 1\times10^{-2}\}, \text{tol} \in \{10^{-5}, 10^{-6}\}.$$

2. **Convection–reaction systems:**

$$n_0 \in \{30,\ 50,\ 70,\ 80,\ 100\}, h \in \{5\times10^{-3},\ 1\times10^{-2}\}, \text{tol} \in \{10^{-5},\ 10^{-6}\}.$$

These ranges span both moderate and relatively stiff regimes, ensuring coverage of the spectral scaling and discretization granularity relevant to practical large-scale problems.

Remark on offline training. All generated systems are used exclusively for training. This offline supervised setup is well-suited to benchmark studies and repeated simulations over a fixed problem family. Its main limitation is the assumption that the training distribution is representative: in time-varying settings such as moving-horizon control with drifting dynamics, the learned mapping may no longer be reliable, and a lightweight online adaptation layer would be needed. We discuss this direction further in Section 5.

Algorithm 1 Data generation for learning-assisted Krylov iteration prediction

Input: Operator families $\mathcal{F}$; grid resolutions $\mathcal{N}$; step sizes $\mathcal{H}$; tolerances $\mathcal{T}$; maximum Krylov dimension $m_{\max}$; random seeds.

Output: Dataset $\mathcal{D}$ of pairs $(\mathbf{x},\, m_{\text{true}})$.

1. Initialize $\mathcal{D} \leftarrow \emptyset$.

2. **For each** $F \in \mathcal{F}$ and $n_0 \in \mathcal{N}$:

a. Construct A and generate low-rank factors B, C.

b. **For each** $(h, \text{tol}) \in \mathcal{H} \times \mathcal{T}$:

i. Compute the feature vector

$$\mathbf{x} \;=\; \big(n_0,\; h,\; \text{tol},\; \|A\|_F,\; \kappa(A),\; \rho(A),\; \texttt{PDE\_ID}\big).$$

ii. Run EBA–BDF(2); increment m until

$$\|R_m(T_f)\|_F \;\le\; \text{tol} \qquad \text{or} \qquad m = m_{\max}.$$

iii. Set $m_{\text{true}} \leftarrow m$.

iv. Append $(\mathbf{x},\, m_{\text{true}})$ to $\mathcal{D}$.

3. Standardize features (see Section 2.4) and finalize $\mathcal{D}$.

Figure 1 summarizes the three-stage workflow of the proposed framework and clarifies how the learning component interacts with the numerical solver. In the first stage, spectral and discretization-based descriptors are extracted from A before any integration is performed. These features are then passed to the trained Random Forest regressor, which outputs a predicted subspace dimension m_{pred}. In the final stage, this prediction is used to initialize the EBA–BDF solver directly, bypassing the trial-and-error phase that would otherwise be required.

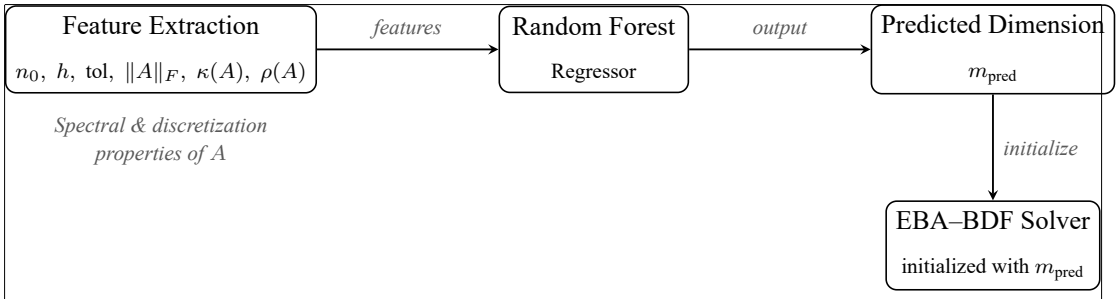

Figure 1: Workflow of the learning-assisted prediction framework: spectral and discretization features are extracted from A, passed to the Random Forest regressor, and the predicted Krylov dimension m_{pred} is used to initialize the EBA–BDF solver.

2.3 Feature Extraction and Target Definition

The target variable for the regression model is the number of Arnoldi iterations m required by EBA–BDF(2) to satisfy the convergence criterion. All remaining descriptors extracted from each simulation are treated as input features. The feature set is not constructed exhaustively; instead, we focus on quantities that are inexpensive to compute prior to integration and that bear a direct connection to classical Krylov convergence theory.

Basic parameters. The grid resolution n_0, time step h, and tolerance tol encode the discretization level and accuracy requirement. These variables are included because m is empirically sensitive to both: finer tolerances systematically require larger subspaces, and coarser grids may alter the spectral scaling of A in ways that affect convergence.

Spectral descriptors. To characterize the operator, we include the Frobenius norm $\|A\|_F$, the induced 1- and ∞-norms, the spectral radius $\rho(A)$, the condition number $\kappa(A)$, and the trace of A. These quantities capture complementary aspects of spectral structure: $\|A\|_F$ and $\rho(A)$ reflect overall scale, $\kappa(A)$ measures sensitivity to perturbations, and the trace provides a coarse summary of eigenvalue distribution.

We also include composite features $N\,\rho(A)$, $\rho(A)/\kappa(A)$, and $\|A\|_F/N$, which encode interactions between problem size and spectral scaling, where $N = n_0^2$ is the total system dimension. These combinations are motivated by classical Krylov convergence bounds [6, 20], which relate the required subspace dimension to the spectral distribution of A: quantities closely linked to eigenvalue clustering — such as spectral radius, conditioning, and operator scaling — are precisely those that govern residual decay rates in projection-based solvers.

Nonnormality. Spectral scale alone does not fully determine convergence behavior. Matrices with similar norms may exhibit markedly different Krylov convergence due to nonnormality, a

phenomenon well documented in convection-dominated problems [21]. To capture this effect, we include the indicator

$$\eta_A = \frac{\|A^T A - A A^T\|_F}{\|A\|_F^2},$$

which measures deviation from normality and serves as a proxy for eigenvector non-orthogonality. In our experiments, η_A is systematically larger for convection–reaction operators and correlates with the cases where m is hardest to predict — confirming that nonnormality captures transient effects not explained by eigenvalues alone. Table 1 reports representative values for each operator family.

Table 1: Representative values of the nonnormality indicator η_A for each operator family.

Family	**Operator**	n_0	N	η_A
Example 1	Diffusion-dominated	40	1600	4.74×10^{-4}
		60	3600	1.62×10^{-4}
		80	6400	7.38×10^{-5}
Example 2	Convection–reaction	30	900	1.01×10^{-3}
		50	2500	3.59×10^{-4}
		70	4900	1.74×10^{-4}

Sparsity. A sparsity measure — the fraction of zero entries in A — is included to reflect discretization structure. While its direct influence on m is secondary compared to spectral descriptors, it helps the model distinguish between operator families with different stencil patterns, particularly at coarser resolutions where sparsity levels vary more noticeably.

Feature selection. The final feature set is determined through a combination of correlation analysis and permutation-based importance measures. This step removes redundant descriptors and stabilizes the learning process — a necessary precaution given the moderate dataset size. The retained features collectively allow the model to distinguish between diffusion-dominated and convection–reaction regimes and to generalize across variations in discretization parameters and problem size.

Physics-informed convergence proxy. Beyond global descriptors, we adopt a residual-decay perspective to further ground the feature set in Krylov theory. In projection-based solvers, the required subspace dimension is directly linked to how rapidly the projected residual decreases with m: operators with well-clustered spectra yield fast, near-geometric decay, while nonnormal operators produce slower and less predictable reduction [20]. The selected features — particularly $\rho(A)$, $\kappa(A)$, and η_A — jointly encode the factors that govern this decay, providing a physically grounded basis for the regression task.

2.4 Preprocessing and Regression Setup

All input features are standardized to zero mean and unit variance prior to training. This step is necessary because the selected descriptors span several orders of magnitude — for instance, $\kappa(A)$ can reach 10^6 while η_A remains close to zero — and without normalization the learning process would be dominated by large-magnitude features regardless of their predictive relevance.

The target variable m is treated as continuous during training and rounded to the nearest integer at evaluation. Model performance is assessed using the coefficient of determination (R^2) and the mean absolute error (MAE). We additionally report $\mathrm{Acc}(|\Delta m| \leq 1), \Delta m = m_{\text{pred}} - m_{\text{true}}$, the proportion of predictions deviating by at most one iteration from the reference. This tolerance-based metric is practically meaningful: a one-step deviation in m has negligible impact on solver efficiency, whereas larger errors can trigger unnecessary subspace growth or premature termination.

Model selection. We compared three candidate models — linear regression, shallow neural networks, and Random Forests — on a held-out validation subset. Linear regression failed to capture the nonlinear interactions between spectral descriptors and m, particularly in the convection–reaction regime. Shallow neural networks were sensitive to initialization and hyperparameter choice, producing unstable results at the available dataset size of approximately 100 instances. Random Forests, by contrast, handled nonlinear feature interactions robustly, required minimal tuning, and provided interpretable feature-importance scores that could be linked back to Krylov convergence theory. We therefore adopt the Random Forest regressor implemented in Scikit-learn [19].

The iteration count m is kept in its original scale. Normalizing m with respect to N was tested but found to obscure the direct relationship between conditioning, tolerance, and subspace size, making the model harder to interpret and validate against theoretical expectations.

Train–test separation. The split strategy is designed to assess genuine generalization rather than interpolation. Training and test sets are constructed from disjoint combinations of (h, tol) and grid sizes n_0: the model is never evaluated on parameter combinations it has seen during training. This is a stricter requirement than a random split, and reflects the intended deployment scenario where m must be predicted for problem configurations that differ from those used to build the dataset.

Generalization is evaluated on two test sets, with deterministic EBA–BDF(2) solutions serving as reference values:

- **Test Set 1 (convection–diffusion; unseen (h, tol)).**

$$(h = 8 \times 10^{-3},\ \mathrm{tol} = 2.5 \times 10^{-6}), \quad n_0 \in \{40,\ 60,\ 80,\ 100\},$$

and

$$(h = 3 \times 10^{-3},\ \mathrm{tol} = 4 \times 10^{-6}), \quad n_0 \in \{30,\ 50,\ 70,\ 100\}.$$

- **Test Set 2 (convection–reaction; unseen grid sizes).**

$$(h = 5 \times 10^{-3},\ \mathrm{tol} = 10^{-5}) \quad \text{with} \quad n_0 \in \{40,\ 60,\ 90\}.$$

3 Training and Computational Setup

All experiments were conducted on a standard workstation (Intel Core i7 CPU, 24 GB RAM) running Python 3.11 with the `scikit-learn` library [19]. The training phase of the Random Forest regressor required only a few seconds compared to the runtime of the deterministic EBA–BDF(2) integrations, indicating that the additional cost of the learning component is small.

Throughout the experiments, the Krylov subspace dimension is denoted by m. The reference value m_{true} corresponds to the number of Arnoldi iterations obtained from the deterministic solver, while m_{pred} denotes the predicted value.

3.1 Example 1. Variable-Coefficient Convection–Diffusion–Reaction Riccati Problem

This benchmark considers a two-dimensional convection–diffusion–reaction system with spatially varying convection terms, designed to evaluate the robustness and generalization capability of the proposed framework. The continuous model is governed by a variable-coefficient convection–diffusion operator defined on the unit square $\Omega = (0, 1)^2$:

$$\partial_t\, x(t, \xi) = \nu\, \Delta x - \alpha\, x + \boldsymbol{\beta}(\xi) \cdot \nabla x + \mathbf{b}(\xi)^T\, \mathbf{u}(t), \qquad x|_{\partial\Omega} = 0, \tag{2}$$

where $\xi = (\xi_1, \xi_2)$, $\nu = 10^{-2}$ is the diffusion coefficient, and $\alpha = 5$ is the reaction rate. Here $\mathbf{u}(t) = [u_1(t),\ u_2(t)]^T \in \mathbb{R}^2$ and $\mathbf{b}(\xi) = [b_1(\xi),\ b_2(\xi)]^T$ denote the control input and the spatial actuator profiles, respectively. The divergence-free convection field $\boldsymbol{\beta}(\xi) = [\beta_1(\xi),\ \beta_2(\xi)]^T$ is defined as

$$\beta_1(\xi) = \sin(\pi\xi_1)\cos(\pi\xi_2), \qquad \beta_2(\xi) = \cos(\pi\xi_1)\sin(\pi\xi_2),$$

which vanishes on the boundary. This choice is consistent with the imposed homogeneous Dirichlet boundary conditions and avoids artificial boundary layers.

A second-order finite-difference discretization on a uniform grid of size $n_0 \times n_0$ yields a sparse system matrix $A \in \mathbb{R}^{N\times N}$, where the total system dimension is $N = n_0^2$. The discretized operator takes the form

$$A = \nu L + D_{\boldsymbol{\beta}} - \alpha I,$$

where L denotes the discrete Laplacian and $D_{\boldsymbol{\beta}}$ represents a finite-difference approximation of the convective term, implemented consistently with the moderate advection regime considered here. The low-rank input and output matrices are $B \in \mathbb{R}^{N\times 2}$ and $C \in \mathbb{R}^{2\times N}$, chosen with Gaussian random entries to generate representative control and observation directions. The spatial discretization leads to a linear time-invariant system of the form

$$\dot{x}(t) = Ax(t) + Bu(t), \qquad y(t) = Cx(t),$$

from which the associated differential Riccati equation is obtained.

This leads to the differential Riccati equation

$$\dot{X}(t) = A^T X(t) + X(t)A - X(t)BB^T X(t) + C^T C, \qquad X(0) = Z_0 Z_0^T,$$

where $Z_0 \in \mathbb{R}^{N\times r}$. This equation was integrated over $t \in [0, 1]$ using the low-rank EBA–BDF(2) method. For each grid resolution, deterministic solutions were computed for multiple timestep (h) and tolerance (tol) combinations, forming the supervised dataset used to train the regression model.

Generalization performance. The model is evaluated on previously unseen parameter combinations $(h, \text{tol}) = (8 \times 10^{-3}, 2.5 \times 10^{-6})$. Table 2 compares deterministic and predicted subspace dimensions.

The predicted values closely match the deterministic references across all matrix sizes, achieving $R^2 = 0.9856$ and MAE $= 4.5$ iterations. Notably, the predictor adapts to changes in both matrix size and discretization parameters without requiring problem-specific tuning, suggesting that the learned mapping captures the dominant spectral factors governing m.

Table 2: Comparison of deterministic and predicted Krylov subspace dimensions under generalization parameters ($h = 8 \times 10^{-3}$, tol $= 2.5 \times 10^{-6}$).

Matrix Size (N)	m_{true}	m_{pred}	T_{true} (s)
1600 (40×40)	148	152.92	264.69
3600 (60×60)	60	65.24	12.87
6400 (80×80)	60	64.00	15.50
10000 (100×100)	60	64.00	15.63

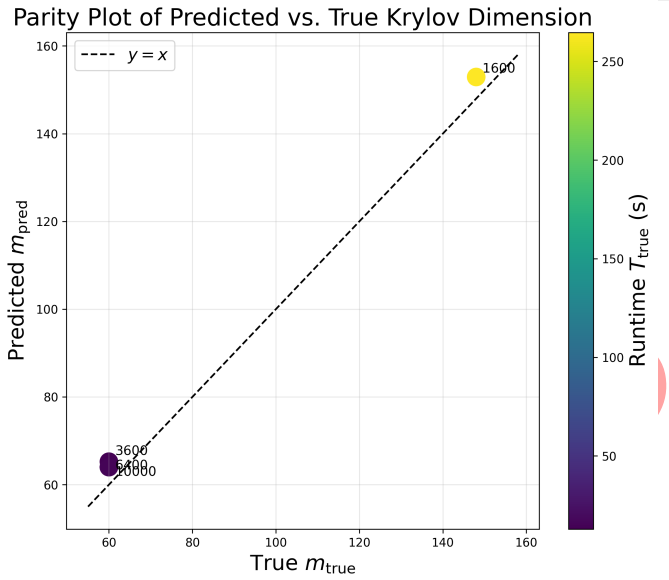

Figure 2: Parity plot of predicted and true Krylov subspace dimensions under ($h = 8 \times 10^{-3}$, tol $= 2.5 \times 10^{-6}$).

Computational efficiency. The prediction time is approximately 50 ms, which is negligible compared to the deterministic solver runtime, confirming that the learning component introduces no significant computational overhead.

To further assess robustness under stricter solver parameters, a second experiment was conducted with $(h, \text{tol}) = (3 \times 10^{-3},\ 4 \times 10^{-6})$. Table 3 reports the corresponding results.

Table 3: Comparison of deterministic and predicted Krylov subspace dimensions under ($h = 3 \times 10^{-3}$, tol $= 4 \times 10^{-6}$).

Matrix Size (N)	m_{true}	m_{pred}	T_{true} (s)
900 (30×30)	236	255.48	3334.39
2500 (50×50)	68	75.44	53.49
4900 (70×70)	56	60.48	28.13
10000 (100×100)	60	64.00	34.89

This apparent inconsistency is explained by the fact that, for the smallest grid (30×30), the required Krylov dimension is relatively large compared to the system size ($m/N \approx 0.26$), placing this case outside the low-rank regime where EBA–BDF operates most efficiently. This effect is specific to low-dimensional settings, where the EBA–BDF method operates outside its optimal regime.

The predictive accuracy remains comparable under stricter discretization parameters, with $R^2 = 0.9795$ and MAE $= 3.25$ iterations. We observe that the model generalizes consis-

tently across previously unseen parameter settings, suggesting that the learned mapping is not restricted to specific discretization regimes.

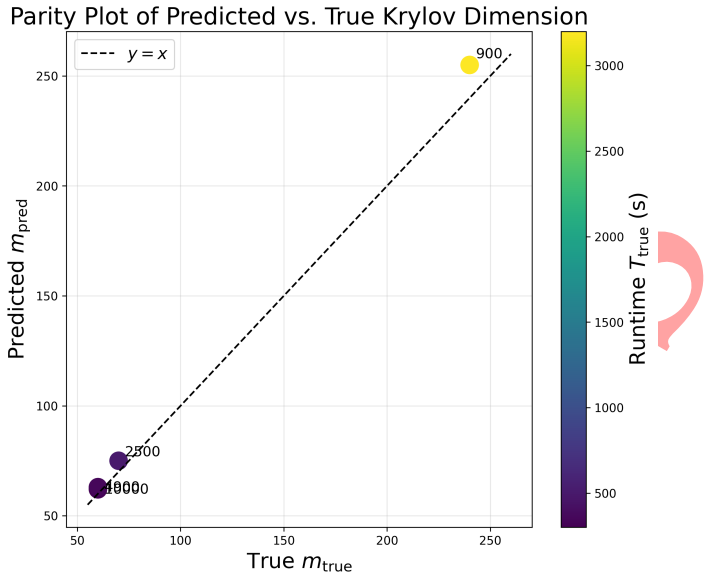

Figure 3: Parity plot of predicted (m_{pred}) and true (m_{true}) Krylov subspace dimensions under $h = 3 \times 10^{-3}$, tol $= 4 \times 10^{-6}$.

Across both parameter regimes, the predictor maintains $R^2 > 0.97$, confirming that the learned mapping between spectral descriptors and m remains stable under parameter changes — including strictly unseen discretization settings.

3.2 Example 2. Two-Dimensional Convection–Diffusion–Reaction Riccati Problem

This example considers a two-dimensional convection–diffusion–reaction system in which advection dominates the dynamics, leading to a more challenging spectral structure for Krylov-based solvers. Although a weak diffusion term is present, the spectral behavior is primarily governed by the interplay between reaction and persistent convective drift, leading to moderately stiff dynamics and nonnormal operator behavior.

The governing PDE is defined on the unit square $\Omega = (0,1)^2$ as

$$\partial_t x(t,\xi) = \varepsilon \Delta x - \alpha x + \boldsymbol{\beta} \cdot \nabla x + \mathbf{b}(\xi)^T \mathbf{u}(t), \qquad x(t,\xi)|_{\partial\Omega} = 0, \tag{3}$$

where $\xi = (\xi_1, \xi_2) \in \Omega$, $\Delta = \partial_{\xi_1\xi_1} + \partial_{\xi_2\xi_2}$ is the Laplacian, and $\nabla = [\partial_{\xi_1}, \partial_{\xi_2}]^T$ is the gradient operator. The output functional is defined as

$$y(t) = \begin{bmatrix} \int_\Omega c_1(\xi)\, x(t,\xi)\, d\xi \\ \int_\Omega c_2(\xi)\, x(t,\xi)\, d\xi \end{bmatrix} \in \mathbb{R}^2.$$

The coefficients are chosen as $\varepsilon = 10^{-2}$ (diffusion), $\alpha = 2$ (reaction rate), and the convective velocity vector is set to $\boldsymbol{\beta} = (0.5, 0.5)^T$, representing a diagonal flow. Compared with Example 1, this configuration maintains the same dimensionality structure but exhibits increased spectral stiffness due to advection dominance. As a consequence, the resulting system matrix exhibits pronounced nonnormality, which is known to slow down and irregularize Krylov convergence.

A second-order finite-difference discretization on a uniform grid of size $n_0 \times n_0$ yields the semi-discrete system

$$\dot{x}(t) = A\,x(t) + B\,u(t), \qquad y(t) = C\,x(t),$$

where the state dimension is $N = n_0^2$. The system matrix $A = \varepsilon\,L + D_{\boldsymbol{\beta}}^{\text{up}} - \alpha\,I$ combines the discrete Laplacian L, the upwinded advection operator $D_{\boldsymbol{\beta}}^{\text{up}}$ (implemented to ensure stability in the advection-dominated regime), and the reaction term.

The input matrix $B \in \mathbb{R}^{N\times 2}$ represents point controls located near $\xi^{(1)} \approx (0.25,\ 0.25)$ and $\xi^{(2)} \approx (0.75,\ 0.75)$. The output matrix $C \in \mathbb{R}^{2\times N}$ is defined by the discretized spatial weight functions:

$$\begin{aligned} c_1(\xi) &= \sin(\pi\xi_1)\,\sin(\pi\xi_2), \\ c_2(\xi) &= \exp\left[-80\left((\xi_1 - 0.5)^2 + (\xi_2 - 0.5)^2\right)\right], \end{aligned}$$

representing a spectral mode and a Gaussian window centered at the domain center $(0.5, 0.5)$, respectively.

The corresponding differential Riccati equation

$$\dot{X}(t) = A^T X(t) + X(t)\,A - X(t)\,B\,B^T X(t) + C^T C, \qquad X(0) = Z_0\,Z_0^T,$$

where $Z_0 \in \mathbb{R}^{N\times r}$. This equation was integrated over $t \in [0, 1]$ using the low-rank EBA–BDF(2) method.

Training and Validation Setup. To rigorously evaluate the model's ability to generalize to unseen problem dimensions, the training and testing datasets were generated using disjoint sets of grid resolutions. The supervised training data was constructed from simulations with spatial grid resolutions $n_0 \in \{30,\ 50,\ 70,\ 80,\ 100\}$, using timesteps $h \in \{5 \times 10^{-3},\ 1 \times 10^{-2}\}$ and tolerances $\text{tol} \in \{10^{-5},\ 10^{-6}\}$. Each deterministic run required at most $m_{\max} = 80$ Arnoldi vectors. The predictive performance was then tested on a separate set of *unseen grid sizes* $n_0 \in \{40,\ 60,\ 65,\ 75,\ 90\}$, which corresponds to matrix dimensions not present in the training phase.

Predictive performance. Table 4 reports deterministic and predicted iteration counts for unseen grid sizes.

The predicted iteration counts closely follow the deterministic references, with deviations typically limited to one or two Arnoldi steps. The regression model achieves $R^2 \approx 0.74$ and MAE $= 1.8$.

The reduced accuracy compared to Example 1 is consistent with the increased spectral complexity of the operator. Nevertheless, the predicted subspace dimensions provide a reasonable initialization for the solver.These findings show that nonnormality and spectral irregularity influence Krylov convergence, in line with classical theory.

Table 4: Comparison of deterministic and predicted Krylov subspace dimensions for unseen grid resolutions ($h = 5 \times 10^{-3}$, tol $= 1 \times 10^{-5}$).

Matrix Size (N)	m_{true}	m_{pred}	T_{true} (s)
1600 (40×40)	28	29	2.82
3600 (60×60)	28	32	2.97
4225 (65×65)	32	32	3.22
5625 (75×75)	32	34	2.97
8100 (90×90)	40	38	10.72

Prediction time (≈ 50 ms) remains negligible compared to the deterministic solver runtime, confirming that the learning component introduces no significant computational overhead.

4 Results and Discussion

This section evaluates the performance of the proposed learning-assisted framework, in which a Random Forest regressor is used to estimate the Krylov subspace dimension m required by the EBA–BDF(2) integrator. The main objective is to assess whether the predicted values provide a reliable and practically useful initialization across different problem settings.

4.1 Predictive Accuracy and Generalization

Across all benchmark problems, the predicted iteration counts closely match the deterministic references and follow the same scaling with respect to problem size. The close match across both examples confirms that the model identifies the dominant factors governing m.

For the diffusion-dominated configurations of Example 1, high accuracy is obtained ($R^2 = 0.9856$), and remains stable under more restrictive solver parameters ($R^2 = 0.9795$). From a

practical standpoint, this level of accuracy is sufficient, since small deviations in m have only a marginal impact on the overall computational cost.

In the convection–reaction setting of Example 2, the predictive accuracy decreases ($R^2 \approx 0.79$), reflecting the increased spectral complexity and stronger nonnormality of the operator. In this regime, Krylov convergence becomes less regular and cannot be fully characterized by global spectral descriptors alone.

Nevertheless, the predicted dimensions provide a reasonable solver initialization — an outcome that reflects the well-known sensitivity of Krylov convergence to nonnormality and spectral irregularity.

To assess generalization, the model is evaluated on discretization parameters and grid sizes that are not included in the training set. The consistent performance across unseen configurations demonstrates that the learned mapping between spectral descriptors and m generalizes beyond the training distribution — across both discretization parameters and problem size.

4.2 Convergence Behavior and Efficiency

In all test cases, the residual norm decreases monotonically as m increases, consistent with the monotonicity property of extended Krylov subspaces. In diffusion-dominated problems, the observed decay is close to geometric, which aligns with the convergence behavior discussed in Section 2.1.

The prediction time is approximately 50 ms, which is negligible compared to the deterministic solver runtime.

Since the prediction time is negligible compared to the solver runtime, the additional overhead of the learning component remains minimal.

4.3 Feature Analysis and Physical Interpretation

Feature-importance analysis indicates that the prediction of m is primarily influenced by spectral and conditioning-related quantities of the matrix A, such as $\|A\|_F$, $\rho(A)$, and $\kappa(A)$, while discretization parameters (h, tol, n_0) play a secondary role.

This observation is consistent with classical Krylov theory, where spectral scaling and conditioning determine how efficiently the subspace captures the dominant system dynamics. Diffusion-dominated operators, which exhibit smoother spectra, typically require smaller subspaces, whereas convection–reaction systems tend to require larger values of m due to increased nonnormal effects.

The alignment between the learned feature ranking and classical Krylov theory is not coincidental: the model assigns highest importance to precisely those quantities — $\|A\|_F$, $\rho(A)$, $\kappa(A)$ — that analytical convergence bounds identify as primary drivers of subspace growth, effectively acting as a nonlinear surrogate for classical convergence theory without requiring problem-specific calibration.

Limitations. The proposed approach relies on global spectral descriptors of the matrix A. While these quantities are effective in diffusion-dominated settings, they may fail to capture localized spectral features and transient growth phenomena that arise in strongly nonnormal operators.

This limitation becomes particularly evident in the convection–reaction regime of Example 2, where matrices with similar spectral descriptors require different Krylov dimensions due to increased nonnormality — for instance, in Example 2, the 60×60 and 90×90 grids share similar $\rho(A)$ values yet require $m_{\text{true}} = 28$ and $m_{\text{true}} = 40$, respectively. Such discrepancies highlight the influence of nonnormality beyond what is captured by global descriptors.

In such settings, the relationship between spectral descriptors and the required Krylov dimension may change over time, which in turn limits the reliability of a purely static offline predictor.

In addition, the current framework is based on an offline supervised learning setup, where the training data is generated from a predefined family of problems. While suitable for repeated simulations, this limits applicability in scenarios where system parameters evolve over time, such as moving-horizon control problems with time-varying dynamics. In such cases, a lightweight online or adaptive learning mechanism would be required.

Finally, the model does not explicitly incorporate analytical properties of the Riccati equation or known error bounds for Krylov approximations. Incorporating such physics-informed features, for example through classical Krylov error bounds (e.g., see [6, 20]), represents a promising direction to improve robustness, particularly in stiff or highly nonnormal regimes where purely data-driven descriptors may be insufficient.

5 Conclusion and Future Work

This work presented a learning-assisted pre-solver for estimating the Krylov subspace dimension m required by low-rank EBA–BDF integrators prior to time integration. The central contribution is the demonstration that m can be reliably inferred from a compact set of spectral and algebraic descriptors — namely spectral radius, conditioning number, and degree of nonnormality — whose relevance is firmly grounded in classical Krylov convergence theory. In

diffusion-dominated regimes, the proposed predictor attains $R^2 > 0.97$, while predictive accuracy decreases to $R^2 \approx 0.79$ in convection–reaction settings, where pronounced nonnormality induces spectral irregularities that global descriptors alone cannot fully resolve.

From a computational standpoint, the offline training cost is incurred once. Thereafter, predictions introduce negligible overhead (≈ 50 ms), confirming that the learning component adds no significant computational cost.The primary benefit is the automation of subspace dimension selection: the predictor eliminates the need for manual tuning of $m_{\max}$, providing an informed initialization without requiring repeated solver runs.The numerical core of EBA–BDF remains unchanged throughout.

A key finding of this study is that classical Krylov convergence theory and data-driven modeling are complementary rather than competing paradigms. The spectral quantities that analytical bounds identify as the primary drivers of subspace growth coincide with those assigned the highest importance by the Random Forest model. This alignment provides principled justification for physics-informed machine learning as a viable and interpretable strategy for automating parameter selection in large-scale matrix equation solvers.

Future Work. Three directions merit further investigation. First, predictive accuracy in stiff and highly nonnormal regimes could be substantially improved by incorporating physics-informed features such as the Péclet and Damköhler numbers, which characterize transport- and reaction-dominated dynamics more precisely than global spectral descriptors. Second, augmenting the feature set with residual-based indicators extracted from the early stages of the Arnoldi process — consistent with classical Krylov error estimates [6, 20] — would enable the model to refine its predictions using partial solver information, thereby enhancing robustness without compromising the lightweight character of the framework. Third, and most broadly, extending the approach toward online adaptation — wherein the predicted m serves as an informed initial estimate that is progressively refined during time integration — would considerably widen its applicability to time-varying settings, including moving-horizon optimal control problems with drifting system dynamics.

Declarations

Availability of Supporting Data

All data generated or analyzed during this study are included in this published paper.

Funding

The author conducted this research without any funding, grants, or support.

Conflict of Interest

The author declares that they have no known competing financial interests or personal relationships that could have influenced the work reported in this paper.

Artificial Intelligence Statement

Artificial intelligence (AI) tools, including large language models, were used solely for language editing and improving readability. AI tools were not used for generating ideas, performing analyses, interpreting results, or writing the scientific content. All scientific conclusions and intellectual contributions were made exclusively by the authors.

Publisher's Note

The publisher remains neutral regarding jurisdictional claims in published maps and institutional affiliations.

References

[1] Abou-Kandil, H., Freiling, G., Ionescu, V., Jank, G. (2003). "Matrix Riccati equations in control and systems theory". *Birkhäuser Basel*. https://doi.org/10.1007/978-3-0348-8081-7

[2] Arias, E., Hernández, V., Ibáñez, J.J., Peinado, J. (2007). "A fixed point-based BDF method for solving differential Riccati equations". *Applied Mathematics and Computation*, 188(2), 1319–1333. https://doi.org/10.1016/j.amc.2006.11.001

[3] Benner, P., Mena, H. (2004). "BDF methods for large-scale differential Riccati equations". *Proceedings of the 16th International Symposium on Mathematical Theory of Networks and Systems*, 1–12. https://csc.mpi-magdeburg.mpg.de/mpcsc/benner/pub/BM_mtns04.pdf

[4] Benner, P., Kürschner, P., Saak, J. (2013). "Efficient handling of complex shift parameters in the low-rank Cholesky-factor ADI method". *Numerical Algorithms*, 62(2), 225–251. https://doi.org/10.1007/s11075-012-9569-7

[5] Benner, P., Li, J.-R., Penzl, T. (2008). "Numerical solution of large-scale Lyapunov equations, Riccati equations, and linear-quadratic optimal control problems". *Numerical Linear Algebra with Applications*, 15(9–10), 755–777. https://doi.org/10.1002/nla.622

[6] Botchev, M.A., Grimm, V., Hochbruck, M. (2013). "Residual, restarting, and Richardson iteration for the matrix exponential". *SIAM Journal on Scientific Computing*, 35(3), A1376–A1397. https://doi.org/10.1137/110820191

[7] Breiman, L. (2001). "Random forests". *Machine Learning*, 45(1), 5–32. https://doi.org/10.1023/A:1010933404324

[8] Chegeni, M.S., Yarahmadi, M. (2025). "Optimal control of linear singularly perturbed systems via eigenvalue assignment". *Control and Optimization in Applied Mathematics*, 10(1), 1–17. https://doi.org/10.30473/coam.2025.73999.1294

[9] Chen, J. (2025). "Graph neural preconditioners for iterative solutions of sparse linear systems". *International Conference on Learning Representations*. https://doi.org/10.48550/arXiv.2406.00809

[10] Chen, H., Dong, H., Geng, Z., Kuang, Y., Luo, J., Wang, H., Wang, J. (2024). "Neural Krylov iteration for accelerating linear system solving". *Advances in Neural Information Processing Systems*, 37, 128636–128667. https://doi.org/10.52202/079017-4086

[11] Dieci, L., Eirola, T. (1994). "Positive definiteness in the numerical solution of Riccati differential equations". *Numerische Mathematik*, 67(3), 303–313. https://doi.org/10.1007/s002110050030

[12] Druskin, V., Knizhnerman, L.A. (1998). "Extended Krylov subspaces: Approximation of the matrix square root and related functions". *SIAM Journal on Matrix Analysis and Applications*, 19, 755–771. https://doi.org/10.1137/S0895479895292400

[13] Druskin, V., Simoncini, V. (2011). "Adaptive rational Krylov subspaces for large-scale dynamical systems". *Systems & Control Letters*, 60(8), 546–560. https://doi.org/10.1016/j.sysconle.2011.04.013

[14] Güldoğan, Y., Hached, M., Jbilou, K., Kurulay, M. (2018). "Low rank approximate solutions to large-scale differential matrix Riccati equations". *Applicationes Mathematicae*, 45(2), 233–254. https://doi.org/10.4064/am2355-1-2018

[15] Hached, M., Jbilou, K., Güldoğan, Y. (2025). "A review on the numerical resolution of large-scale low-rank differential matrix equations". In: *Jbilou, K., Hached, M., Maniar, L., Ratnani, A. (eds) Mathematical Modeling with Modern Applications. M3A 2024. Springer Proceedings in Mathematics & Statistics, vol 497. Springer, Cham*, 151–180. https://doi.org/10.1007/978-3-031-89041-3_8

[16] Häusner, P., Öktem, O., Sjölund, J. (2024). "Neural incomplete factorization: Learning preconditioners for the conjugate gradient method". *Transactions on Machine Learning Research*. https://doi.org/10.48550/arXiv.2305.16368

[17] Lancaster, P., Rodman, L. (1995). "Algebraic Riccati equations". *Oxford: Clarendon Press / Oxford University Press*. https://doi.org/10.1093/oso/9780198537953.001.0001

[18] Li, Y., Chen, P.Y., Du, T., Matusik, W. (2023). "Learning preconditioners for conjugate gradient PDE solvers". *Proceedings of the 40th International Conference on Machine Learning, PMLR*, 202, 19425–19439. https://proceedings.mlr.press/v202/li23e.html

[19] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., VanderPlas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, É. (2011). "Scikit-learn: machine learning in Python". *Journal of Machine Learning Research*, 12(85), 2825–2830. http://jmlr.org/papers/v12/pedregosa11a.html

[20] Saad, Y. (2003). "Iterative methods for sparse linear systems", 2nd ed., *Society for Industrial and Applied Mathematics*. https://doi.org/10.1137/1.9780898718003

[21] Simoncini, V. (2016). "Computational methods for linear matrix equations". *SIAM Review*, 58(3), 377–441. https://doi.org/10.1137/130912839

Author Bio-sketch

Yaprak Güldoğan Dericioğlu received the B.Sc. and M.Sc. degrees in mathematics from İnönü University, Türkiye, and the Ph.D. degree in applied mathematics from Yildiz Technical University, Istanbul, Türkiye, in 2019. She is currently an Assistant Professor in the Department of Mathematics at Bitlis Eren University, Türkiye. Her research interests include numerical linear algebra, differential matrix equations, imbalanced learning, and machine learning–assisted numerical methods. She has worked on data reduction techniques for complex imbalanced datasets and physics-informed learning frameworks. Email: yguldogan@beu.edu.tr