

# Analysis of Truncated Singular Value Decomposition for Koopman Operator-Based Lane Change Model

Chinnawut Nantabut<sup>1a</sup>

*The Sirindhorn International Thai-German Graduate School of Engineering, King Mongkut's University of Technology  
North Bangkok, 1518 Pracharat 1 Road, Wongsawang, Bangsue, Bangkok, Thailand*

**Keywords:** Automated Driving, System Identification, Lane Change Model, Koopman Operator, Truncated Singular Value Decomposition.

**Abstract:** Understanding and modeling complex dynamic systems is crucial for enhancing vehicle performance and safety, especially in the context of autonomous driving. Recently, popular methods such as Koopman operators and their approximators, known as Extended Dynamic Mode Decomposition (EDMD), have emerged for their effectiveness in transforming strongly nonlinear system behavior into linear representations. This allows them to be integrated with conventional linear controllers. To achieve this, Singular Value Decomposition (SVD), specifically truncated SVD, is employed to approximate Koopman operators from extensive datasets efficiently. This study evaluates different basis functions used in EDMD and ranks for truncated SVD for representing lane change behavior models, aiming to balance computational efficiency with information loss. The findings, however, suggest that the technique of truncated SVD does not necessarily achieve substantial reductions in computational training time and results in significant information loss.

## 1 INTRODUCTION

In automotive engineering, model or system identification is crucial for understanding the behavior of traffic participants under various driving conditions. This understanding is essential for improving vehicle performance and safety, particularly when transferring insights from scene analysis in these scenarios to autonomous vehicles, enhancing their decision-making accuracy.


Given the complexity of modeling such behaviors, the underlying dynamic systems are inherently nonlinear. To address these challenges, various modeling techniques are employed, particularly black-box models, which do not require explicit logical or physical representations of the relationship between inputs and outputs. Examples of these data-driven approaches include neural networks, series models, and autoregressive models (Mauroy and Goncalves, 2020).

One of the prominent data-driven approaches is the Koopman operator. Although initially developed long ago (Koopman, 1931), it has gained renewed attention in modern applications for capturing the behavior of dynamical systems. Its appeal lies in its

connection to classical methods, the ability to integrate measurement-based formulations suitable for machine learning, and the potential for simplification in real-world applications (Brunton et al., 2021). Another advantage is that it provides a global representation of the system, unlike instantaneous linearization or dynamic linearization, which offer only a local approximation of the model around a specific operating point.

Since Koopman operators are theoretically infinite-dimensional, the Extended Dynamic Mode Decomposition (EDMD) method is employed to approximate them using a set of smaller, more manageable basis functions.

Koopman operators have been widely applied to model identification, particularly within the automotive industry (Manzoor et al., 2023). Notable examples include (Cibulka et al., 2019), where a single-track model derived from a twin-track model was used to generate trajectory data by varying tire forces and kinematic variables. Their findings showed that increased model complexity does not necessarily improve accuracy. Similarly, (Yu et al., 2022a) assessed model fidelity in lane-change scenarios using various systems, while (Yu et al., 2022b) demonstrated the ability of Koopman operators to reduce compu-

<sup>a</sup>  <https://orcid.org/0000-0002-5767-6023>

tational complexities in vehicle tracking across different road types. Moreover, (Kim et al., 2022) applied Koopman operators to model lane-keeping systems by defining states based on lateral dynamics and tire models, utilizing a Linear Quadratic Regulator (LQR) for control. Their follow-up work (Kim et al., 2023) compared multiple basis functions for model identification, introducing systematic selection methods and signal normalization techniques. Meanwhile, (Joglekar et al., 2023) investigated function approximators for Koopman operators in path-tracking applications.

The operators, particularly when paired with model predictive control (MPC), have gained traction for transforming nonlinear models into linear forms, enhancing controller performance as shown by (Abraham et al., 2017). (Gupta et al., 2022) addressed eco-driving for electric vehicles using the Koopman operator with simulator data for high-performance MPC. Similarly, (Buzhardt and Tallapragada, 2022) generated trajectories using a half-car and a terramechanics model in deformable off-road scenarios, applying the Koopman operator to lift states like vertical displacement and using MPC for regulation.

Recent studies highlight the increasing use of neural networks as function approximators for the Koopman operator. For example, (Han et al., 2020) demonstrated the integration of Koopman operators with MPC, contrasting it with reinforcement learning approaches (RL). Similarly, (Xiao et al., 2023) enhanced long-term predictability by combining deep neural networks with Koopman operators, utilizing real driving data. In another study, (Guo et al., 2023) focused on lane-changing scenarios with shared control systems, forming the Koopman operator through neural networks and comparing the resulting paths to real driver trajectories. Additionally, (Chen et al., 2024a) employed neural networks to reformulate states into input constraints for MPC, while (Bongiovanni et al., 2024) applied the Koopman operator to model the behavior of electrical throttle valves. Innovatively, (Chen et al., 2024b) used neural network-based autoencoders for approximating driving behavior and implemented adaptive MPC to handle parameter variations, showcasing the growing synergy between neural networks and Koopman operators in automotive applications.

As mentioned earlier, Koopman operators enable a linear representation of dynamic systems, which can be framed in a linear state-space formulation using a system matrix. To identify this matrix from the available data, it is necessary to invert a matrix constructed from these data values. Given that this matrix is typically large and asymmetric, Singular Value Decom-

position (SVD) is utilized to enhance numerical stability. Additionally, due to the substantial size of the data, the approximation technique known as truncated SVD is employed to reduce the dimensions of the matrices involved in the SVD process while preserving as much essential information as possible from the original inverted matrix. These methodologies not only expedite the training process required to derive the system matrix but also come with the trade-off of potential information loss.

Procedures like truncated SVD have not been thoroughly explored in the literature, with similar investigations primarily found in (Wilson, 2023), which aimed to tackle the overfitting problem. However, this study did not address the systematic approximation rank of the matrix, and its application was outside the realm of automotive contexts, concentrating instead on simple dynamic models. This gap highlights the need for further research into the utilization of truncated SVD specifically within automotive applications, where the complexities of dynamic systems demand more robust identification techniques.

In this paper, the use of truncated SVD for approximating the system matrix derived from the basis functions utilized in EDMD is investigated, as discussed in Section 2. The focus is centered on a use case involving lane-changing behavior. Initially, data is generated using simple geometric models, followed by the introduction and selection of basis functions that serve as approximators. The process of calculating system dynamics through SVD is detailed, highlighting the significance of the truncated version and the procedure for selecting the appropriate matrix rank. All implementations are conducted in Python, as outlined in Section 3. Finally, the resulting approximations are analyzed and discussed in comparison to the original matrix, considering aspects such as information loss and time complexity.

It is important to note that topics like MPC as a controller are not within the scope of this work and are therefore excluded. Unlike the majority of the literature reviewed, the primary focus is on system or model identification through the application of Koopman operators, alongside the associated techniques of EDMD and truncated SVD.

## 2 METHOD

The analysis presented in this paper is summarized in Figure 1. First, a simplified vehicle model for lane change behavior is explained in Section 2.1, leading to the generation of trajectories denoted as  $\mathcal{T}$ . Next, the primary focus of this paper - system identification

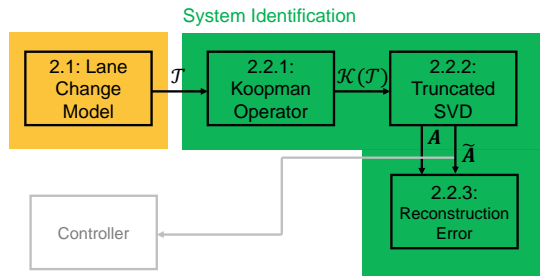


Figure 1: Analysis steps for the truncated SVD-based Koopman operator in a lane change model.

- is executed as detailed in Section 2.2, which consists of three steps.

The Koopman operator  $\mathcal{K}$  and the EDMD, along with their basis functions  $\Phi$ , are introduced in Section 2.2.1. The transformed trajectories  $\mathcal{K}(\mathcal{T})$  or their approximators  $\Phi(\mathcal{T})$  are then used to determine the system matrix  $\mathbf{A}$ , which describes the linear system.

Subsequently, an appropriate truncated SVD, selected based on specific criteria, is discussed in Section 2.2.2, resulting in the approximated system matrix  $\tilde{\mathbf{A}}$ . While the model for these trajectories is typically employed for controller design, this aspect is not addressed in the current paper. Finally, the original matrix  $\mathbf{A}$  and the approximated one  $\tilde{\mathbf{A}}$  are compared to form the reconstruction errors, as discussed in Section 2.2.3.

## 2.1 Lane Change Model

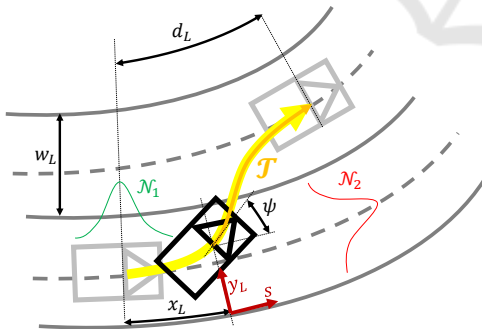


Figure 2: The calculation of lane change trajectories  $\mathcal{T}$  is based on (Schreier, 2017).

Since, in this paper, real data were not collected to approximate the model described by the Koopman operator, a mathematically simplified model proposed by (Schreier, 2017) is used instead, as depicted in Figure 2.

Additionally, the longitudinal and lateral movement of the vehicle can be modeled independently. The vehicle transitions from the right to the left lane of width  $w_L$  in the Frenet-Serret frame, also known as

the lane frame, described by the coordinates  $s$  and  $y_L$ . The  $s$  axis runs along the right edge of the right lane, whereas the  $y_L$  axis is perpendicular to it and directed to the left.

Furthermore, the vehicle is located at the position  $(x_L, y_L)$  and is oriented to the lane at an angle  $\psi$ . The longitudinal length of the complete trajectory, depicted in yellow, is defined by  $d_L$ . Starting from the current pose, including the vehicle's position and orientation, a trajectory  $\mathcal{T}$  in orange is generated until the vehicle reaches the middle line of the left lane.

To model the longitudinal movement, a constant acceleration model is utilized. In this model, the longitudinal state at time step  $k$  is defined as  $\mathbf{X}_{s,k} = [s_k \ v_{s,k} \ a_{s,k}]^T$ , where  $s_k$  denotes the longitudinal displacement along the road,  $v_{s,k}$  represents the longitudinal velocity, and  $a_{s,k}$  indicates the longitudinal acceleration at time step  $k$ . The next state  $\mathbf{X}_{s,k+1}$  can be sampled from the normal distribution  $\mathcal{N}_1$ , as shown in green in Figure 2:

$$\begin{bmatrix} s_{k+1} \\ v_{s,k+1} \\ a_{s,k+1} \end{bmatrix} \sim \mathcal{N} \left( \begin{bmatrix} 1 & T & \frac{1}{2}T^2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_k \\ v_{s,k} \\ a_{s,k} \end{bmatrix}, \begin{bmatrix} \frac{1}{4}T^4 & \frac{1}{2}T^3 & \frac{1}{2}T^2 \\ \frac{1}{2}T^3 & T^2 & T \\ \frac{1}{2}T^2 & T & 1 \end{bmatrix} \sigma_{a_s}^2 \right), \quad (1)$$

where the first argument of the expression denotes the mean vector, and the second one is represented by the covariance matrix that depends on the sample time  $T$  and the standard deviation in the longitudinal acceleration  $\sigma_{a_s}$ .

To model the lateral movement, which is approximated by sinusoidal geometry, the initial lateral position  $y_{L,0}$  is generated from the normal distribution  $\mathcal{N}_2$ , as shown in red in Figure 2:

$$y_{L,0} \sim \mathcal{N}(0.5w_L, \sigma_{y_L}^2), \quad (2)$$

where  $\sigma_{y_L}$  denotes the standard deviation in the lateral displacement.

The start angle  $\psi_0$  can be sampled from a uniform distribution  $\mathcal{U}$ , with the exclusion of zero:

$$\psi_0 \sim \mathcal{U}[0, \psi_{0,max}], \quad (3)$$

where  $\psi_{0,max}$  is the maximal possible initial yaw angle.

As derived in (Schreier, 2017), where  $y_{L,0}$  must also be larger than  $0.5w_L$ , the longitudinal length of the complete trajectory  $d_L$  can then be calculated by:

$$d_L = \frac{w_L \pi}{2 \tan(\psi_0)} \cos \left( \sin^{-1} \left( \frac{2y_{L,0}}{w_L} - 2 \right) \right), \quad (4)$$

as well as the current longitudinal displacement  $x_L$ :

$$x_L = \left( \frac{1}{2} + \frac{1}{\pi} \sin^{-1} \left( \frac{2y_{L,0}}{w_L} - 2 \right) \right) d_L. \quad (5)$$

The initial position is then defined by  $(x_L, y_{L,0})$ , where the longitudinal position at time step  $k = 0$  is set to  $s_0 = x_L$ . The longitudinal position  $s_k$  is updated using the constant acceleration model. The new lateral position  $y_{L,k}$  is then computed by:

$$y_{L,k} = \frac{w_L}{2} \sin \left( \frac{\pi}{d_L} (s_k + x_L) - \frac{\pi}{2} \right) + w_L. \quad (6)$$

Subsequently, the trajectories for the following system identification  $\mathcal{T}$  can be defined by a set of  $N_T$  trajectories  $\mathcal{T}_i$ , denoted by:

$$\mathcal{T}_i = \bigcup_{k=0}^{k_{max}} \{(s_k, y_{L,k})_i\}, \quad (7)$$

where  $k_{max} = \max_k \{k \mid (s_k + x_L) \leq d_L\}$ .

## 2.2 System Identification

Once the trajectories are generated in Section 2.1, the next step is to understand how the system evolves based on the presented data. Given the assumption that the system is complex and nonlinear, Koopman operators and EDMD are employed to elevate the system into a higher-dimensional space where it can be described linearly. This transformation enables the application of typical linear modeling techniques such as LQR or linear MPC.

Section 2.2.1 elaborates on the Koopman operator and its approximators, EDMD, highlighting their roles in system transformation. Subsequently, to optimize computational efficiency, Section 2.2.2 discusses the reduction of linear system dynamics using truncated SVD, focusing on methods to select an appropriate threshold.

Finally, Section 2.2.3 covers the methods for reconstructing truncated trajectories and compares them with the original trajectories.

### 2.2.1 Koopman Operator

The concept of the Koopman operator  $\mathcal{K}$  is to transform or "lift" the original nonlinearly dependent adjacent states to another space where the relationship between the new or lifted states becomes linear, as illustrated in Figure 3. Here, the original states are described by trajectories obtained from the previous section, consisting of tuples representing longitudinal displacement  $s$  and the lateral displacement  $y_L$ .

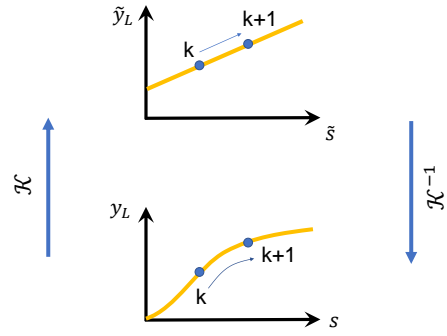


Figure 3: The Koopman operator  $\mathcal{K}$  "lifts" states at different time steps, from  $(s, y_L)$  to  $(\tilde{s}, \tilde{y}_L)$ . This transformation converts the nonlinear dependency among adjacent states into a linear one. Dealing with the lifted states becomes more manageable compared to the original states.

Since these trajectories are generated from sinusoidal curves, the transition from one state to another is inherently nonlinear. With the Koopman operator  $\mathcal{K}$ , the states can be transformed into the lifted ones  $\mathcal{K}(s, y_L)$  such that

$$\mathcal{K}(s, y_L)_{k+1} = \mathbf{A} \mathcal{K}(s, y_L)_k, \quad (8)$$

where  $\mathbf{A}$  is the system matrix describing the linear dependency between the current (at time step  $k$ ) lifted states  $(\tilde{s}, \tilde{y}_L)_k = \mathcal{K}(s, y_L)_k$ , and the next lifted states  $(\tilde{s}, \tilde{y}_L)_{k+1} = \mathcal{K}(s, y_L)_{k+1}$ .

Since the Koopman operator is infinite-dimensional, it can only be approximated using EDMD. (8) can then be transformed into:

$$\Phi(s, y_L)_{k+1} \approx \mathbf{A} \Phi(s, y_L)_k, \quad (9)$$

where  $\Phi$  denotes the function approximator that aims to describe the linearity as accurately as possible.

There are various options for selecting function approximators. A common approach involves using a set of basis functions to construct them. One widely used type is the monomial basis:

$$\Phi_m(s, y_L) = \begin{bmatrix} s \\ y_L \\ s^2 \\ y_L^2 \\ \vdots \\ s^{N_m} \\ y_L^{N_m} \end{bmatrix}, \quad (10)$$

where  $N_m$  denotes the highest order of polynomials.

Another common type is the thin plate spline radial basis, described by:

$$\Phi_r(s, y_L) = \begin{bmatrix} s \\ y_L \\ \|s - c_s\|_2^2 \log(\|s - c_s\|_2) \\ \|y_L - c_y\|_2^2 \log(\|y_L - c_y\|_2) \end{bmatrix}, \quad (11)$$

where  $\|\cdot\|_2$  denotes the  $l^2$ -norm (which, in this context, is simply the absolute value), and the constants  $c_s$  as well as  $c_y$  are randomly chosen.

At the end, the transformed trajectory  $\mathcal{K}(\mathcal{T}_i)$  is obtained, which can be approximated as:

$$\Phi(\mathcal{T}_i) = \bigcup_{k=0}^{k_{max}} \{\Phi(s_k, y_{L,k})_i\}. \quad (12)$$

### 2.2.2 Truncated SVD

In the case of EDMD, once the states are lifted using the operator  $\Phi$ , the lifted states are then stored and utilized for subsequent training processes. To determine the system matrix  $\mathbf{A}$  that captures the linearity, the inversion of  $\mathcal{K}(s, y_L)_k$  in (8) or  $\Phi(s, y_L)_k$  in (9) is necessary. This involves employing SVD for numerical stability, which can also be truncated to expedite the training process. However, not only two tuples of  $\Phi(s, y_L)_k$  are utilized, but the entire trajectories are used to determine the system matrix  $\mathbf{A}$ . At first, a so-called snapshot matrix of trajectory  $\mathcal{T}_i$ ,  $\mathbf{X}_i$ , can be defined as:

$$\mathbf{X}_i = [\Phi(s_0, y_{L,0})_i \quad \dots \quad \Phi(s_{k_{max}-1}, y_{L,k_{max}-1})_i], \quad (13)$$

and its right-shifted snapshot matrix  $\mathbf{X}'_i$  is denoted by:

$$\mathbf{X}'_i = [\Phi(s_1, y_{L,1})_i \quad \dots \quad \Phi(s_{k_{max}}, y_{L,k_{max}})_i]. \quad (14)$$

Consequently, the total snapshot matrix of all trajectories  $\mathcal{T}$ ,  $\mathbf{X}$ , is computed by:

$$\mathbf{X} = [\mathbf{X}_1 \quad \mathbf{X}_2 \quad \dots \quad \mathbf{X}_{N_r}], \quad (15)$$

and its total shifted snapshot matrix  $\mathbf{X}'$  is denoted by:

$$\mathbf{X}' = [\mathbf{X}'_1 \quad \mathbf{X}'_2 \quad \dots \quad \mathbf{X}'_{N_r}]. \quad (16)$$

The linear dynamics of the system can be approximated, using the system matrix  $\mathbf{A}$ , by:

$$\mathbf{X}' \approx \mathbf{A}\mathbf{X}, \quad (17)$$

analogous to (8) and (9).

Since the matrix  $\mathbf{X}$  is not square, its pseudo-inverse or Moore-Penrose inverse  $\mathbf{X}^\dagger$  is computed instead to calculate the system matrix  $\mathbf{A}$ :

$$\mathbf{A} \approx \mathbf{X}'\mathbf{X}^\dagger, \quad (18)$$

where  $\mathbf{X}^\dagger = \mathbf{X}^T(\mathbf{X}\mathbf{X}^T)^{-1}$  as derived from the so-called normal equation.

Alternatively, the snapshot *real* matrix  $\mathbf{X}$  can be reformulated using SVD, resulting in the form:

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T, \quad (19)$$

which results in the pseudo-inverse  $\mathbf{X}^\dagger$  of the form:

$$\mathbf{X}^\dagger = \mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^T, \quad (20)$$

where the  $\mathbf{\Sigma}$ -matrix is displayed as:

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_1 & 0 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & \sigma_2 & 0 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \dots & \sigma_{r_{max}} & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 0 & \dots & 0 \end{bmatrix}, \quad (21)$$

given that this matrix (and therefore  $\mathbf{X}$ ) has a rank of  $r_{max}$ , the singular values are sorted in descending order, namely  $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_{r_{max}}$ .

By choosing a specific rank  $r \leq r_{max}$ , an approximated  $\tilde{\mathbf{X}}$  in the truncated SVD is obtained by:

$$\tilde{\mathbf{X}} = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^T, \quad (22)$$

where  $\tilde{\mathbf{U}} = \mathbf{U}_{:,1:r}$ ,  $\tilde{\mathbf{\Sigma}} = \mathbf{\Sigma}_{1:r,1:r}$ , and  $\tilde{\mathbf{V}} = \mathbf{V}_{1:r,:}$ .

Now, the question of how to systematically choose the rank  $r$  is posed. As suggested in (Brunton and Kutz, 2019), the percentual accumulated "energy" up to rank  $r$  can be calculated as:

$$E_r = 100\% \cdot \frac{\sum_{i=1}^r \sigma_i}{\sum_{i=1}^{r_{max}} \sigma_i}. \quad (23)$$

Typically, the empirical values of  $E_r = 90\%$  or  $E_r = 99\%$  are used to determine the appropriate rank, resulting in a good representation  $\tilde{\mathbf{X}}$  of the original snapshot matrix  $\mathbf{X}$ . Alternatively, a hard threshold (HT) based on the ratio  $\beta = \frac{m}{n}$  of the dimensions of the (snapshot) matrix  $\mathbf{X}_{n \times m}$  and the median of its singular values  $\sigma_{med}$  is recommended by (Gavish and Donoho, 2014). The hard threshold  $r_{HT}$  is calculated by:

$$r_{HT} = \omega(\beta)\sigma_{med}, \quad (24)$$

where  $\omega(\beta)$  is based purely on the ratio  $\beta$  and can be calculated accordingly. Since some values of the relationship between  $\beta$  and  $\omega$  are provided, interpolation techniques can be employed to roughly determine the hard threshold rank  $r_{HT}$  used to approximate the snapshot matrix  $\mathbf{X}$ .

The system matrix  $\mathbf{A}$  can be approximated by an approximated system matrix  $\tilde{\mathbf{A}}$ :

$$\tilde{\mathbf{A}} = \mathbf{X}'\tilde{\mathbf{V}}\tilde{\mathbf{\Sigma}}^{-1}\tilde{\mathbf{U}}^T. \quad (25)$$

Analogous to (9), the approximated system dynamics can then be described as:

$$\Phi(s, y_L)_{k+1} \approx \tilde{\mathbf{A}}\Phi(s, y_L)_k. \quad (26)$$

The next states  $(s, y_L)_{k+1}$  can then be retrieved by inverting the operator  $\Phi$ .



### 2.2.3 Reconstruction Error

To evaluate model fidelity, the Frobenius norm  $\|\mathbf{B}\|_F$  of a matrix  $\mathbf{B}$  can be used, where  $\|\mathbf{B}\|_F = \sqrt{\sum_j \sum_i b_{ij}^2}$  and  $b_{ij}$  denotes the elements of the matrix  $\mathbf{B}$ . This norm allows for a comparison of the system dynamics between the full-ranked system matrix  $\mathbf{A}$  and its truncated version  $\tilde{\mathbf{A}}$ .

The relative reconstruction error of the truncated matrix  $\tilde{\mathbf{A}}$  is represented by:

$$RE_{\tilde{\mathbf{A}}} = \frac{\|\mathbf{A} - \tilde{\mathbf{A}}\|_F}{\|\mathbf{A}\|_F}. \quad (27)$$

The time consumption of the truncated matrix  $\tilde{\mathbf{A}}$  can also be measured relative to the full matrix  $\mathbf{A}$ :

$$\tilde{t}_{\tilde{\mathbf{A}}} = \frac{t_{\tilde{\mathbf{A}}}}{t_{\mathbf{A}}}. \quad (28)$$

All the steps outlined in Figure 1 are summarized in Algorithm 1 for the easy implementation of the analysis proposed in this paper. This includes the definition of necessary variables, the collection of data, the execution of truncated SVD, and the analysis of the results using various metrics.

## 3 SIMULATION

All parameters utilized in the previous section, as summarized in Algorithm 1, are defined here. The comparison between the original data and the Koopman operators using different types of basis functions is illustrated. Additionally, qualitative trajectories based on truncated SVD with various rank selections are compared and analyzed. Finally, the time consumption for these processes is also discussed.

### 3.1 Setup

Taken from the original paper for modeling lane change behavior (Schreier, 2017), the lane width  $w_L$  is set to 3.5 m. With the vehicle width  $w_V$  of 1.5 m, the standard deviation of lateral deviation  $\sigma_{y_L} = \frac{1}{3}0.5(w_L - w_V) = \frac{1}{3}$  m, whereas the value of the longitudinal direction  $\sigma_{a_s} = \frac{0.2}{3} \frac{m}{s^2}$ . The time constant  $T$  is set to 0.1 s. The initial kinematic variables are  $s_0 = 0$  m,  $v_0 = 10 \frac{m}{s}$ , and  $a_0 = 0 \frac{m}{s^2}$ . The vehicle can start with a maximal orientation of  $\psi_0 = 15^\circ$ .

To ensure both basis functions have the same dimension,  $N_m$  is set to 2. Furthermore, the constants  $c_s$  and  $c_y$  are sampled from a uniform distribution ranging between  $-\frac{w_L}{2}$  and  $+\frac{w_L}{2}$ .

Step I: Define all parameters:

1. Lane Change Model:
  - (a) Road geometry  $w_L$
  - (b) Standard deviations:  $\sigma_{a_s}$ , and  $\sigma_{y_L}$
  - (c) Time constant  $T$
  - (d) Kinematic variables  $a_0$ ,  $v_0$ ,  $s_0$ , and  $\psi_{0,max}$
2. Basis function:
  - (a) Monomial  $N_m$
  - (b) Thin plate spline radial  $c_s$  and  $c_y$

Step II: Collecting data: Trajectory puffer  $\mathcal{T}$

For each trajectory  $i$ ,  $\mathcal{T}_i \leftarrow \emptyset$ :

1. Sample the initial lateral displacement  $y_{L,k} = y_{L,0}$  from (2) and the initial yaw angle  $\psi_0$  from (3)
2. Calculate the trajectory's longitudinal length  $d_L$  from (4) and the current displacement  $s_k = x_L$  from (5)
3.  $\mathcal{T}_i \leftarrow \{(s_k, y_{L,k})\}$
4. While  $s_k \leq d_L$ :
  - (a) Update the longitudinal displacement  $s_k$  using (1) and the lateral displacement  $y_{L,k}$  using (6)
  - (b)  $\mathcal{T}_i \leftarrow \mathcal{T}_i \cup \{(s_k, y_{L,k})\}$
5. Update  $\mathcal{T} \leftarrow \mathcal{T} \cup \mathcal{T}_i$

Step III: Truncated SVD

1. Calculate snapshot matrices  $\mathcal{X}$  and  $\mathcal{X}'$  based on (13), (14), (15) and (16)
2. Use SVD on the snapshot matrix  $\mathcal{X}$  using (19)
3. Calculate the rank  $r$  using (23) or (24)
4. Construct truncated snapshot matrix  $\tilde{\mathcal{X}}$  using (22) and the system matrix  $\tilde{\mathbf{A}}$  using (25)

Step IV: Evaluation

1. Calculate the relative reconstruction error  $RE_{\tilde{\mathbf{A}}}$  using (18), (25) and (27) and the relative time consumption  $\tilde{t}_{\tilde{\mathbf{A}}}$  using (28)

Algorithm 1: Truncated SVD for Koopman Operator-Based Lane Change Model.

### 3.2 Results

In Figure 4, a single trajectory illustrating a lane change maneuver (from the red right middle lane to the green left middle lane) is depicted in blue, based on the approach by (Schreier, 2017) as detailed in Section 2.1. The vehicle follows a simple sinusoidal pattern, returning to the right lane after exceeding the maximum longitudinal displacement  $d_L$ , as previously discussed.

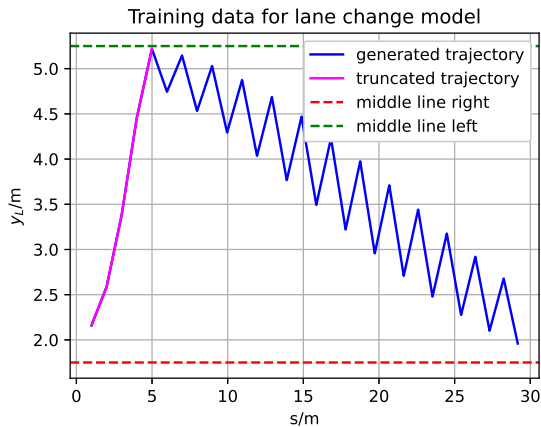


Figure 4: An exemplary original trajectory used for training the system matrix  $\mathbf{A}$  or  $\tilde{\mathbf{A}}$  illustrates a disadvantage of choosing this model when the longitudinal displacement ( $s_k + x_L$ ) reaches the maximum longitudinal sinusoidal length  $d_L$ , as explained in Equation 7.

Subsequently, strong oscillations occur, stemming from the constant acceleration model used for longitudinal movement, impacting the lateral movement as well. Therefore, the generated data needs truncation before further model training. However, this truncation results, as shown in magenta, in shorter trajectories than anticipated, potentially leading to model overfitting.

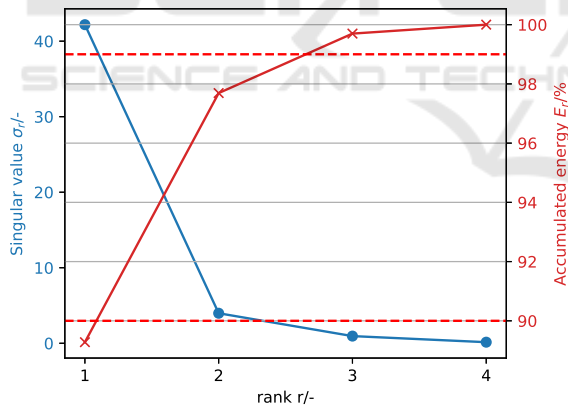


Figure 5: The sorted singular values  $\sigma_r$  and their accumulated energy  $E_r$  are visualized against their ranks  $r$ , using the example of monomial basis functions. Horizontal lines representing the empirical values  $E_r = 90\%$  and  $E_r = 99\%$  are also plotted to aid in identifying their corresponding singular value.

Now, the truncated trajectory is subjected to truncated SVD by constructing the matrix  $\mathbf{X}$  and sorting its singular values  $\sigma_i$ . The accumulated energy  $E_r$  is calculated for each singular value, and their values are plotted in Figure 5 using monomial basic functions. In this example, it is observed that approximately  $E_r = 90\%$  is achieved directly at the first sin-

gular value ( $E_r \approx 89\%$ ), as it is relatively large ( $\geq 30$ ) compared to the others. According to the plot, the third singular value corresponds to  $E_r = 99\%$ .

However, upon calculating the hard threshold rank  $r_{HT}$ , it is found that it exceeds 4. Therefore, the full rank is used in this case instead. Additionally, similar observations have been made for the radial basis functions, resulting in identical rank selections.

Table 1: Comparison of the truncated SVD performance ( $E_r = 90\%$ ,  $E_r = 99\%$  and  $HT$ : hard threshold) based on different basis functions ( $m$ : monomial and  $r$ : thin plate spline radial), including metrics such as the relative reconstruction error and the time consumption.

Basis	rank $r$	$RE_{\tilde{\mathbf{A}}}/\%$	$\min\{\tilde{t}_{\tilde{\mathbf{A}}}\}/\%$
$\Phi_{m,90\%}$	1	99.54	89.97
$\Phi_{m,99\%}$	3	95.76	96.47
$\Phi_{m,HT}$	4 (full)	0	100
$\Phi_{r,90\%}$	1	98.14	95.16
$\Phi_{r,99\%}$	3	60.81	97.21
$\Phi_{r,HT}$	4 (full)	0	100

The relative reconstruction error  $RE_{\tilde{\mathbf{A}}}$  and the *minimal* relative time consumption  $\min\{\tilde{t}_{\tilde{\mathbf{A}}}\}$  of the system matrix  $\mathbf{A}$  are compared in Table 1. It is observed for each type of basis function that the reconstruction error decreases with higher rank selections.

Additionally, in the lane change model with the previously collected data, the radial basis function demonstrates superior model fidelity. However, with truncated SVD, the reconstruction errors are relatively large ( $> 50\%$ ), prompting the use of the full-ranked matrix  $\mathbf{A}$  due to the hard threshold.

Furthermore, truncated SVD does not consistently reduce time consumption, as evidenced by cases where  $\tilde{t}_{\tilde{\mathbf{A}}}$  exceeds 100%. The *minimal* time consumption is highlighted here, underscoring the potential for reduced computation time with truncated SVD.

## 4 CONCLUSIONS

This study investigates the efficacy of truncated SVD in conjunction with Koopman operators and EDMD for system identification in the context of lane change behavior. The results indicate that while these techniques do not necessarily lead to a significant reduction in the computational time required for training the system matrix, they entail a compromise in model fidelity.

To validate these findings, future research will explore the use of diverse datasets for training and conduct a more comprehensive statistical analysis, as well as analyze controllers not covered here.

## REFERENCES

- Abraham, I., Torre, G. D. L., and Murphey, T. D. (2017). Model-based control using koopman operators. *CoRR*, abs/1709.01568.
- Bongiovanni, N., Mavkov, B., Martins, R., and Allibert, G. (2024). Data-Driven Nonlinear System Identification of a Throttle Valve Using Koopman Representation. In *American Control Conference (ACC 2024)*, Toronto, Canada.
- Brunton, S. L., Budišić, M., Kaiser, E., and Kutz, J. N. (2021). Modern koopman theory for dynamical systems.
- Brunton, S. L. and Kutz, J. N. (2019). *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press, USA, 1st edition.
- Buzhardt, J. and Tallapragada, P. (2022). A koopman operator approach for the vertical stabilization of an off-road vehicle. *IFAC-PapersOnLine*, 55(37):675–680. 2nd Modeling, Estimation and Control Conference MECC 2022.
- Chen, H., He, X., Cheng, S., and Lv, C. (2024a). Deep koopman operator-informed safety command governor for autonomous vehicles. *IEEE/ASME Transactions on Mechatronics*, pages 1–11.
- Chen, Z., Chen, X., Liu, J., Cen, L., and Gui, W. (2024b). Learning model predictive control of nonlinear systems with time-varying parameters using koopman operator. *Applied Mathematics and Computation*, 470:128577.
- Cibulka, V., Hanis, T., and Hromcik, M. (2019). Data-driven identification of vehicle dynamics using koopman operator. *2019 22nd International Conference on Process Control (PC19)*, pages 167–172.
- Gavish, M. and Donoho, D. L. (2014). The optimal hard threshold for singular values is  $4/\sqrt{3}$ . *IEEE Transactions on Information Theory*, 60(8):5040–5053.
- Guo, W., Zhao, S., Cao, H., Yi, B., and Song, X. (2023). Koopman operator-based driver-vehicle dynamic model for shared control systems. *Applied Mathematical Modelling*, 114:423–446.
- Gupta, S., Shen, D., Karbowski, D., and Rousseau, A. (2022). Koopman model predictive control for eco-driving of automated vehicles. In *2022 American Control Conference (ACC)*, pages 2443–2448.
- Han, Y., Hao, W., and Vaidya, U. (2020). Deep learning of koopman representation for control. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 1890–1895.
- Joglekar, A., Samak, C., Samak, T., Kosaraju, K. C., Smereka, J., Brudnak, M., Gorsich, D., Krovi, V., and Vaidya, U. (2023). Analytical construction of koopman edmd candidate functions for optimal control of ackermann-steered vehicles. *IFAC-PapersOnLine*, 56(3):619–624. 3rd Modeling, Estimation and Control Conference MECC 2023.
- Kim, J. S., Quan, Y. S., and Chung, C. C. (2022). Data-driven modeling and control for lane keeping system of automated driving vehicles: Koopman operator approach. In *2022 22nd International Conference on Control, Automation and Systems (ICCAS)*, pages 1049–1055.
- Kim, J. S., Quan, Y. S., and Chung, C. C. (2023). Koopman operator-based model identification and control for automated driving vehicle. *International Journal of Control, Automation and Systems* 21, page 2431–2443.
- Koopman, B. O. (1931). Hamiltonian systems and transformation in hilbert space. *Proceedings of the National Academy of Sciences*, 17(5):315–318.
- Manzoor, W. A., Rawashdeh, S., and Mohammadi, A. (2023). Vehicular applications of koopman operator theory—a survey. *IEEE Access*, 11:25917–25931.
- Mauroy, A. and Goncalves, J. (2020). Koopman-based lifting techniques for nonlinear systems identification. *IEEE Transactions on Automatic Control*, 65(6):2550–2565.
- Schreier, M. (2017). Bayesian environment representation, prediction, and criticality assessment for driver assistance systems. *at - Automatisierungstechnik*, 65(2):151–152.
- Wilson, D. (2023). Koopman operator inspired nonlinear system identification. *SIAM Journal on Applied Dynamical Systems*, 22(2):1445–1471.
- Xiao, Y., Zhang, X., Xu, X., Liu, X., and Liu, J. (2023). Deep neural networks with koopman operators for modeling and control of autonomous vehicles. *IEEE Transactions on Intelligent Vehicles*, 8(1):135–146.
- Yu, S., Shen, C., and Ersal, T. (2022a). Autonomous driving using linear model predictive control with a koopman operator based bilinear vehicle model. *IFAC-PapersOnLine*, 55(24):254–259. 10th IFAC Symposium on Advances in Automotive Control AAC 2022.
- Yu, S., Sheng, E., Zhang, Y., Li, Y., Chen, H., and Hao, Y. (2022b). Efficient nonlinear model predictive control of automated vehicles. *Mathematics*, 10(21).