
NONINTRUSIVE MODEL ORDER REDUCTION FOR CROSS-DIFFUSION SYSTEMS

A PREPRINT

✉ **Bülent Karasözen**

Institute of Applied Mathematics & Department of Mathematics
Middle East Technical University, Ankara-Turkey
bulent@metu.edu.tr

✉ **Murat Uzunca**

Department of Mathematics
Sinop University, Sinop-Turkey
muzunca@sinop.edu.tr

✉ **Güden Mülayim**

Department of Mathematics
Adıyaman University, Adıyaman-Turkey
Institute of Applied Mathematics
Middle East Technical University, Ankara-Turkey
gulden.mulayim@metu.edu.tr

ABSTRACT

In this paper, we investigate tensor based nonintrusive reduced-order models (ROMs) for parametric cross-diffusion equations. The full-order model (FOM) consists of ordinary differential equations (ODEs) in matrix or tensor form resulting from finite-difference discretization of the differential operators by taking the advantage of Kronecker structure. The matrix/tensor differential equations are integrated in time with the implicit-explicit (IMEX) Euler method. The reduced bases, relying on a finite sample set of parameter values, are constructed in form of a two-level approach by applying higher-order singular value decomposition (HOSVD) to the space-time snapshots in tensor form, which leads to a large amount of computational and memory savings. The nonintrusive reduced approximations for an arbitrary parameter value are obtained through tensor product of the reduced basis by the parameter dependent core tensor that contains the reduced coefficients. The reduced coefficients for new parameter values are computed using radial basis function (RBF) interpolation. The efficiency of the proposed method is illustrated through numerical experiments for two-dimensional Schnakenberg and three-dimensional Brusselator cross-diffusion equations. The spatiotemporal patterns are accurately predicted by the reduced-order models with speed-up factors of orders two and three over the full-order models.

Keywords Pattern formation, implicit-explicit methods, matrix differential equations, Sylvester equation, reduced order modelling, tensor algebra

1 Introduction

Reaction-diffusion systems have been largely employed in literature to predict spatiotemporal patterns occurring in biological sciences, chemistry and physics. The correlation between diffusion and cross-diffusion terms may cause unstable steady solutions in form of patterns like labyrinths, spots, stripes, etc. These patterns may exhibit dynamical behavior like oscillation, annihilation, aggregation, segregation, and replication in a long time. The common aspect of pattern formation is the interplay between diffusion and reaction, known as diffusion-driven or Turing instability. A generalization of diffusion-driven instability is the cross-diffusion, which is characterized by a gradient in the density of one species inducing a flux in direction of another species. Reaction-diffusion models which take into account the effects of self-diffusion as well as cross-diffusion are widely used to describe spatiotemporal dynamics of many two interacting species [1, 2, 3, 4, 5, 6, 7, 8, 9]. In contrast to the classical reaction-diffusion systems without cross-

diffusion, it is no longer necessary to enforce that one of the species diffuses much faster than the other for the occurrence of spatiotemporal patterns [5].

Cross-diffusion systems are coupled systems of semi-linear partial differential equations (PDEs). They have been discretized in space by various methods like finite differences, finite volumes, and finite elements. In order to resolve the patterns accurately, very fine meshes in space and time are needed in numerical simulations. The effects of cross-diffusion on pattern formation in reaction-diffusion systems have been studied theoretically and numerically in many papers. Numerical simulations require fine spatial grids and long-term integration. Furthermore, multi-query simulations are required for the prediction of patterns in the parameter space. A Cross-diffusion system involves many parameters limiting the use of standard vector-based ordinary differential equation (ODE) solvers in time because of excessive computational costs in two and three-dimensional domains. Under certain assumptions on the domain, one can take advantage of the Kronecker structure arising in standard space discretizations of the differential operators, and the resulting system of ODEs can be treated directly in matrix or tensor form [10, 11, 12, 13]. By exploiting the structure of the diffusion matrix, the matrix/tensor based versions of classical time integrators, such as implicit-explicit (IMEX) methods [12] allow for much finer problem discretizations. They are based on the explicit factorization of small matrices, requiring a sequence of small matrix/tensor problems, i.e., Sylvester equations. Exploiting the spectral structure of these matrices, the computational cost is reduced further. Due to the modest size of these matrices, the computational cost per iteration can be made lower than that of the corresponding vector approaches, by working in the reduced spectral space. In this paper, we employ the matrix/tensor oriented strategy in [12] for space-time discretization of cross-diffusion systems in two and three-dimensional domains.

Simulation of the cross-diffusion systems to predict the spatiotemporal patterns for different parameter combinations take a long time and are computationally very expensive. Reduced-order model (ROM) methods have been developed to reduce the dimension of large dynamic systems. The main idea of ROM is to construct basis functions on low-dimensional reduced space and then project onto a full-order model (FOM) to obtain reduced-order solutions. ROMs for time-dependent parametrized PDEs have to approximate solutions as a function of time, spatial coordinates, and a parameter vector, which turns out to be more challenging. Reduced-order modeling techniques are generally implemented in an offline-online paradigm. In the offline stage, a set of reduced basis functions are extracted from the snapshots, i.e., a collection of high-fidelity solutions, and the reduced basis is computed by combining them. In the online phase, the FOM is projected onto the reduced space that represents the main dynamics of the FOM, and the solutions for new parameters are computed in an efficient manner. Based on the offline-online methodology, ROM methods are classified into two categories: intrusive and nonintrusive ROM methods. The intrusive ROM methods determine the reduced solutions by solving a reduced order model, i.e., a projection of the FOM onto the reduced space. The proper orthogonal decomposition (POD) with the Galerkin projection [14, 15] is one of the most popular tool used in intrusive ROM methods. The POD extracts the reduced basis through the singular value decomposition (SVD) of the snapshot matrix obtained by sampling in parameter space. Then, ROM is constructed by applying Galerkin projection. To handle this problem, some nonlinearity treatment methods are introduced, such as discrete empirical interpolation method (DEIM) [16]. Although all ROM methods are accurate to approximate solutions, they depend on the governing equations and discretized forms of them, that is these methods are intrusive. Another class of ROM methods is the data-driven or nonintrusive ROM (NIROM) methods which are based only on accessing to snapshots and do not use governing equations.

Contrary to a large number of papers for reduced-order modeling of patterns in fluid flows, there are few studies about the prediction of spatiotemporal patterns of reaction-diffusion equations [17, 18]. In this paper, we follow the NIROM approach in [19, 20, 21] which is based on a two-level POD approach by exploiting the matrix/tensor based discretization of the cross-diffusion system. In the first level, the reduced basis is computed by applying the higher-order SVD (HOSVD) [22, 23] to the space-time snapshots related to each parameter value from a sample set of parameter values, instead of using classical SVD as in [19, 20]. In the second level, the global set of reduced bases and coefficients of the reduced solutions are computed. The undetermined coefficients in the approximation are estimated using a nonintrusive approach based on radial basis function (RBF) approximation (in contrast to Galerkin projection). The reduced solution for a new parameter value is then obtained by interpolating the reduced solutions with the RBF. Recently, HOSVD is used as intrusive ROM with POD [24] and as space-time nonintrusive ROM [25]. The matrix based discretization in [12] is exploited in construction of intrusive ROMs [26] with the POD and DEIM. In [26] this approach is generalized to higher-order tensor differential equations in the framework of POD and DEIM with Galerkin projection using HOSVD. In this paper, the ROMs are constructed nonintrusively from space-time full-order solutions in the matrix and tensor forms with the HOSVD. The reduced solutions for new parameter values are computed by utilizing the radial basis function (RBF) interpolation. The numerical experiments on the two-dimensional Schnakenberg and three-dimensional Brusselator cross-diffusion equations show that the patterns are predicted accurately for new parameter values. Using the nonintrusive approach with HOSVD, large amount of

computer memory and computational time is saved, which is observed in high compression rates and speed-up factors of reduced-order solutions over the full-order solutions.

The rest of the paper is organized as follows. In Section 2, we briefly describe the cross-diffusion systems and give the matrix/tensor based space discretization by finite-differences, with the IMEX Euler time integration. The tensor based space-time nonintrusive ROM is presented in Section 3. Numerical results illustrating the accuracy and efficiency of the ROM methodology for the prediction of spatiotemporal patterns are given in Section 4 for two examples of cross-diffusion systems: two-dimensional Schnakenberg and three-dimensional Brusselator equations. The paper ends with some conclusions in Section 5.

2 Full order model

In this section, we briefly introduce the cross-diffusion system and describe the matrix/tensor based discretization in space and time. We use the following notation. Scalars will be denoted as lower-case letters, vectors as bold lower-case letters, matrices are represented by capital letters or bold capital letters, and order- d tensors ($d > 2$) as calligraphic capital letters.

2.1 Cross-diffusion systems

Cross-diffusion systems are characterized by a gradient in the concentration of one species inducing a flux of another chemical species. In nature, cross-diffusion expresses the population fluxes of one species, preys, due to the presence of the other species, predators. The two-component cross-diffusion system is given as

$$\begin{aligned} u_t &= d_u \nabla^2 u + d_{vu} \nabla^2 v + f(u, v), & (\mathbf{x}, t) &\in \Omega \times (0, t_f], \\ v_t &= d_v \nabla^2 v + d_{uv} \nabla^2 u + g(u, v), & (\mathbf{x}, t) &\in \Omega \times (0, t_f], \\ \frac{\partial u}{\partial \mathbf{n}} &= \frac{\partial v}{\partial \mathbf{n}} = 0, & (\mathbf{x}, t) &\in \partial\Omega \times [0, t_f], \\ u(\mathbf{x}, 0) &= u_0(\mathbf{x}), \quad v(\mathbf{x}, 0) = v_0(\mathbf{x}) & \mathbf{x} &\in \Omega, \end{aligned} \quad (1)$$

where $\Omega \in \mathbb{R}^2$ ($\Omega \in \mathbb{R}^3$) is the spatial domain with the boundary $\partial\Omega$, $\mathbf{x} = (x, y)^T \in \Omega$ ($\mathbf{x} = (x, y, z)^T \in \Omega$) is the spatial point, \mathbf{n} is the exterior unit normal vector to the boundary, and $[0, t_f]$ is the time domain for a final time $t_f > 0$. The non-negative bounded functions $u_0(\mathbf{x}) > 0$ and $v_0(\mathbf{x}) > 0$ are prescribed as initial conditions. In the cross-diffusion system (1), the unknown components $u(\mathbf{x}, t)$ and $v(\mathbf{x}, t)$ represent chemical concentrations or population densities. The cross-diffusion system (1) is a semi-linear PDE consisting of the linear diffusion parts with the Laplace operator $\nabla^2 = \partial^2/\partial x^2 + \partial^2/\partial y^2$ ($\nabla^2 = \partial^2/\partial x^2 + \partial^2/\partial y^2 + \partial^2/\partial z^2$), and nonlinear reaction terms $f(u, v)$ and $g(u, v)$. The self-diffusion coefficients $d_u > 0$ and $d_v > 0$ are always positive whereas the cross-diffusion ones d_{vu} and d_{uv} can be either positive or negative. The cross-diffusion coefficient d_{vu} indicates the influence of the density of $v(\mathbf{x}, t)$ to the density of $u(\mathbf{x}, t)$ so that $u(\mathbf{x}, t)$ is repelled from $v(\mathbf{x}, t)$ when $d_{vu} > 0$, or otherwise $u(\mathbf{x}, t)$ is attracted to $v(\mathbf{x}, t)$ when $d_{vu} < 0$. The other cross-diffusion coefficient d_{uv} has the same meaning with the role of $u(\mathbf{x}, t)$ and $v(\mathbf{x}, t)$ are switched. In other words, species with positive cross-diffusion move towards the other species with the lower concentration, while in case of the negative cross-diffusion coefficients the respective species moves towards the higher concentration regime of the other species.

In the cross-diffusion systems of type (1), there exist a variety of reactions terms $f(u, v)$ and $g(u, v)$ with polynomial nonlinearities, such as the Schnakenberg model [4, 5], Brusselator model [2, 3, 6], Gray-Scott model [1]. Many cross-diffusion systems have nonlinear reaction terms in form of the rational functions (see for example [7, 8, 9]). The reaction terms in the cross-diffusion Schnakenberg model [5] are given by

$$f(u, v) = \gamma(\alpha - u + u^2 v), \quad g(u, v) = \gamma(\beta - u^2 v), \quad (2)$$

where γ is a positive constant describing the relative strength of reaction terms. The reaction terms of the Brusselator cross-diffusion system [2, 3, 6] have similar form as the one given in (2) of the Schnakenberg model

$$f(u, v) = -(\beta + 1)u + u^2 v + \alpha, \quad g(u, v) = \beta u - u^2 v. \quad (3)$$

Cross-diffusion systems are also parameter dependent PDEs. In addition to the self-diffusion and cross-diffusion parameters d_u, d_v, d_{uv}, d_{vu} , the nonlinear reaction terms includes parameters such as α, β as given in (2) and (3). In this paper, we study the parameter dependent reduced-order solutions $u(\mathbf{x}, t; \boldsymbol{\mu})$ and $v(\mathbf{x}, t; \boldsymbol{\mu})$ of the system (1) in a parameter space \mathcal{D} . In this section, we suppress the parameter dependency of the states u and v to simplify the notation.

2.2 Matrix and tensor based discretization

Semi-discretization of the cross-diffusion system (1) in space with finite differences, finite elements, and spectral methods inside a hypercube in \mathbb{R}^d ($d = 2, 3$) leads to a system of ODEs in the following form

$$\begin{aligned} \dot{\mathbf{u}} &= d_u A \mathbf{u} + d_{vu} A \mathbf{v} + \mathbf{f}(\mathbf{u}, \mathbf{v}), & \mathbf{u}(0) &= \mathbf{u}_0, \\ \dot{\mathbf{v}} &= d_{uv} A \mathbf{u} + d_v A \mathbf{v} + \mathbf{g}(\mathbf{u}, \mathbf{v}), & \mathbf{v}(0) &= \mathbf{v}_0, \end{aligned} \quad (4)$$

where the entries of the matrix A accounts for the spatial discretization of the diffusion terms including the Laplace operator ∇^2 on a discrete mesh of the domain Ω . The time dependent vectors $\mathbf{u}(t), \mathbf{v}(t) : [0, t_f] \rightarrow \mathbb{R}^N$ are the semi-discrete approximations of the unknown solutions $u(\mathbf{x}, t)$ and $v(\mathbf{x}, t)$ of the system (1), and $\mathbf{f}(\mathbf{u}, \mathbf{v}), \mathbf{g}(\mathbf{u}, \mathbf{v}) : \mathbb{R}^N \times \mathbb{R}^N \rightarrow \mathbb{R}^N$ are the nonlinear vectors corresponding to the nonlinear functions $f(u, v)$ and $g(u, v)$, after spatial discretization. All the state vectors and nonlinear terms are evaluated componentwise at the spatial grid points. Moreover, the number N denotes the degree of freedom of the discrete spatial grid. When the finite-differences are used for the space discretization, for instance, we have $N = n_1 n_2$ ($N = n_1 n_2 n_3$), where n_1 and n_2 (n_1, n_2 and n_3) are the number of the spatial nodes in x and y -directions (x, y and z -directions), respectively.

Most of the time integrators are developed for solving the semi-discretized ODEs in vector form like (4). For an accurate simulation of the patterns of cross-diffusion systems (1), fine spatial discretization is required. This limits the use of standard vector-based ODE solvers in time because of the excessive computational cost and computer memory. By exploiting the structure of the matrix of Laplace operator after space discretization, the semi-discrete ODE system (4) can be written as a matrix/tensor differential equation. The space discretization by means of matrix/tensor based leads to the solution of linear equations with small matrices, which allows to much finer discretization of the problem and reduces the cost of full-order solutions. In this paper, we apply the matrix/tensor based approach in [12] for the solution of the linear cross-diffusion systems (1) in two and three space dimensions.

For finite differences methods, for certain finite elements techniques and spectral methods, the Laplace operator ∇^2 can be discretized by means of a tensor basis. To do this, let the matrix $T_n \in \mathbb{R}^{n \times n}$ given by

$$T_n = \frac{1}{h^2} \begin{pmatrix} -2 & 2 & & & 0 \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & & 1 & -2 & 1 \\ 0 & & & & 2 & -2 \end{pmatrix},$$

denotes the matrix corresponding to the discretization of the Laplace operator by centered finite differences under homogeneous Neumann boundary condition on a one-dimensional spatial mesh (an interval) $\Omega = [0, \ell] \subset \mathbb{R}$ consisting of n grid points including the end points, and with the uniform mesh size $h = \ell/(n-1)$. Then, the discretization of the Laplace operator on a rectangular domain $\Omega = [0, \ell_x] \times [0, \ell_y] \subset \mathbb{R}^2$ leads to a matrix A of the form

$$A = I_{n_2} \otimes T_{n_1} + T_{n_2} \otimes I_{n_1} \in \mathbb{R}^{N \times N}, \quad N = n_1 n_2, \quad (5)$$

whereas it has the form

$$A = I_{n_3} \otimes I_{n_2} \otimes T_{n_1} + I_{n_3} \otimes T_{n_2} \otimes I_{n_1} + T_{n_3} \otimes I_{n_2} \otimes I_{n_1} \in \mathbb{R}^{N \times N \times N}, \quad N = n_1 n_2 n_3, \quad (6)$$

on a rectangular prism $\Omega = [0, \ell_x] \times [0, \ell_y] \times [0, \ell_z] \subset \mathbb{R}^3$. Here, I_{n_1}, I_{n_2} and I_{n_3} are n_1, n_2 and n_3 -dimensional identity matrices, respectively, and \otimes denotes the Kronecker product. The numbers n_1, n_2 and n_3 denote the number of the nodes in x, y and z -directions with the mesh sizes $h_x = \ell_x/(n_1-1)$, $h_y = \ell_y/(n_2-1)$ and $h_z = \ell_z/(n_3-1)$, respectively. Throughout the paper, we simplify the notation by taking $T_1 = T_{n_1}, T_2 = T_{n_2}$ and $T_3 = T_{n_3}$ together with $I_1 = I_{n_1}, I_2 = I_{n_2}$ and $I_3 = I_{n_3}$ with the appropriate dimension.

2.3 Full discretization on two-dimensional domains

We consider a discrete mesh on a rectangular domain $\Omega = [0, \ell_x] \times [0, \ell_y] \subset \mathbb{R}^2$ with the mesh sizes $h_x = \ell_x/(n_1-1)$ and $h_y = \ell_y/(n_2-1)$, and with the grid nodes $\mathbf{x}_{ij} = (x_i, y_j)$, where $x_i = (i-1)h_x$ and $y_j = (j-1)h_y$, $i = 1, \dots, n_1$, $j = 1, \dots, n_2$. In order to represent the time dependent semi-discrete matrix solutions at the grid nodes, we introduce the matrix functions $U(t), V(t) : [0, t_f] \mapsto \mathbb{R}^{n_1 \times n_2}$, which contain the same components of the solution vectors $\mathbf{u}(t)$ and $\mathbf{v}(t)$ in the form $U_{ij}(t) = u(\mathbf{x}_{ij}, t)$ and $V_{ij}(t) = v(\mathbf{x}_{ij}, t)$, respectively. The rows and columns of U and V reflect the space discretization of the given problem in x and y -directions, respectively.

On the other hand, the solution vectors $\mathbf{u}(t)$ and $\mathbf{v}(t)$ in the ODE system (4) can be related to the vectorization of the solution matrices $U(t)$ and $V(t)$ by the $\text{vec}(\cdot)$ operator defined by $\mathbf{u}(t) = \text{vec}(U(t))$ and $\mathbf{v}(t) = \text{vec}(V(t))$,

respectively. With this operation, for instance, each column of the matrix $U(t)$ is stuck one after the other in order to obtain the vector $\text{vec}(U(t))$. This implementation satisfies a lexicographic order of the nodes in the rectangular grid for a finite difference discretization. With this notation and using the properties of the Kronecker product together with the identity (5), we have that $Au = \text{vec}(T_1U + UT_2^T)$ and $Av = \text{vec}(T_1V + VT_2^T)$. Then, the vectorial ODE system (4) can be equivalently written as the following matrix differential equation [12]

$$\begin{aligned}\dot{U} &= d_u(T_1U + UT_2^T) + d_{vu}(T_1V + VT_2^T) + F(U, V), \\ \dot{V} &= d_{uv}(T_1U + UT_2^T) + d_v(T_1V + VT_2^T) + G(U, V),\end{aligned}\quad (7)$$

where the nonlinear matrix functions $F, G : \mathbb{R}^{n_1 \times n_2} \times \mathbb{R}^{n_1 \times n_2} \mapsto \mathbb{R}^{n_1 \times n_2}$ are given by $F_{ij}(U, V) = f(U_{ij}, V_{ij})$ and $G_{ij}(U, V) = g(U_{ij}, V_{ij})$, $i = 1, \dots, n_1$, $j = 1, \dots, n_2$, with the property that $\mathbf{f}(\mathbf{u}, \mathbf{v}) = \text{vec}(F(U, V))$ and $\mathbf{g}(\mathbf{u}, \mathbf{v}) = \text{vec}(G(U, V))$.

Semi-discrete diffusion problems like (4) are stiff problems, which makes explicit methods inappropriate. In the presence of nonlinear reaction terms, fully implicit schemes require a nonlinear solver, e.g., Newton, at each time step. IMEX schemes are splitting methods for ODE systems, where the stiff linear diffusion part is integrated implicitly, while the nonlinear reaction part is integrated explicitly, as a consequence, only one linear system must be solved at each time step. We consider the discrete times $t_k = k\Delta t$, $k = 0, \dots, n_t$, with the time step $\Delta t = t_f/n_t$. The semi-discrete matrix differential equation (7) is solved with IMEX Euler method [12], which leads to the following full discrete system

$$\begin{aligned}\frac{U^{k+1} - U^k}{\Delta t} &= d_u(T_1U^{k+1} + U^{k+1}T_2^T) + d_{vu}(T_1V^{k+1} + V^{k+1}T_2^T) + F(U^k, V^k), \\ \frac{V^{k+1} - V^k}{\Delta t} &= d_{uv}(T_1U^{k+1} + U^{k+1}T_2^T) + d_v(T_1V^{k+1} + V^{k+1}T_2^T) + G(U^k, V^k),\end{aligned}\quad (8)$$

where the full discrete solution matrices are given as $U^k = U(t_k) \in \mathbb{R}^{n_1 \times n_2}$ and $V^k = V(t_k) \in \mathbb{R}^{n_1 \times n_2}$, $k = 0, \dots, n_t$, with the given initial solution matrices U^0 and V^0 satisfying $U_{ij}^0 = u_0(\mathbf{x}_{ij})$ and $V_{ij}^0 = v_0(\mathbf{x}_{ij})$, respectively.

The matrix/tensor formulation (7) has the same convergence and stability properties of the underlying time discretization methods for the classical vector differential equation (4) [12]. Exploiting the structure of the Laplace operator in the linear part and using finer grids, highly accurate full-order solutions are obtained and the computational cost is much reduced. After collecting the alike terms in the full discrete system (8), we obtain a system of linear matrix equation in the form of the Sylvester equation

$$\begin{aligned}(I_1 - d_u\Delta t T_1)U^{k+1} - d_u\Delta t U^{k+1}T_2^T - d_{vu}\Delta t T_1V^{k+1} - d_{vu}\Delta t V^{k+1}T_2^T &= U^k + \Delta t F(U^k, V^k), \\ (I_1 - d_v\Delta t T_1)V^{k+1} - d_v\Delta t V^{k+1}T_2^T - d_{uv}\Delta t T_1U^{k+1} - d_{uv}\Delta t U^{k+1}T_2^T &= V^k + \Delta t G(U^k, V^k),\end{aligned}\quad (9)$$

which is solved for the solution matrices U^{k+1} and V^{k+1} .

The matrix/tensor methods can be made more efficient by computing a-priori spectral decomposition of the coefficient matrices/tensors of not too large sizes [12]. Assuming that the matrices T_1 and T_2^T are diagonalizable, the solution of the Sylvester equation (9) is accelerated by the use of the eigenvalue decomposition of the matrices T_1 and T_2^T . Let the eigenvalue decompositions $T_1 = X\Lambda^{(x)}X^{-1}$ and $T_2^T = Y\Lambda^{(y)}Y^{-1}$ are given, with the matrices $X \in \mathbb{R}^{n_1 \times n_1}$ and $Y \in \mathbb{R}^{n_2 \times n_2}$ of nonsingular vectors, and the diagonal matrices $\Lambda^{(1)} = \text{diag}(\lambda_1^{(1)}, \dots, \lambda_{n_1}^{(1)})$ and $\Lambda^{(2)} = \text{diag}(\lambda_1^{(2)}, \dots, \lambda_{n_2}^{(2)})$ of the eigenvalues $\lambda_i^{(1)}$ and $\lambda_j^{(2)}$, $i = 1, \dots, n_1$, $j = 1, \dots, n_2$. Multiplying both equations in (9) from left by X^{-1} and from right by Y , substituting eigenvalue decompositions $T_1 = X\Lambda^{(1)}X^{-1}$ and $T_2^T = Y\Lambda^{(2)}Y^{-1}$, and setting $\hat{U}^k = X^{-1}U^kY$ and $\hat{V}^k = X^{-1}V^kY$, we reach the system of matrix equations

$$\begin{aligned}(I_1 - d_u\Delta t \Lambda^{(1)})\hat{U}^{k+1} - d_u\Delta t \hat{U}^{k+1}\Lambda^{(2)} - d_{vu}\Delta t \Lambda^{(1)}\hat{V}^{k+1} - d_{vu}\Delta t \hat{V}^{k+1}\Lambda^{(2)} &= Q_1^k, \\ (I_1 - d_v\Delta t \Lambda^{(1)})\hat{V}^{k+1} - d_v\Delta t \hat{V}^{k+1}\Lambda^{(2)} - d_{uv}\Delta t \Lambda^{(1)}\hat{U}^{k+1} - d_{uv}\Delta t \hat{U}^{k+1}\Lambda^{(2)} &= Q_2^k,\end{aligned}\quad (10)$$

where

$$\begin{aligned}Q_1^k &= X^{-1}(U^k + \Delta t F(U^k, V^k))Y, \\ Q_2^k &= X^{-1}(V^k + \Delta t G(U^k, V^k))Y.\end{aligned}\quad (11)$$

The matrix equation (10) in which all the coefficient matrices on the left hand sides are diagonal matrices, can be easily solved componentwise. Therewith, the entries of the solution matrices $U^{k+1}, V^{k+1} \in \mathbb{R}^{n_1 \times n_2}$ are given as the following 2×2 linear system of equations

$$\begin{pmatrix} S_{ij}^{11} & S_{ij}^{12} \\ S_{ij}^{21} & S_{ij}^{22} \end{pmatrix} \begin{pmatrix} \hat{U}_{ij}^{k+1} \\ \hat{V}_{ij}^{k+1} \end{pmatrix} = \begin{pmatrix} (Q_1^k)_{ij} \\ (Q_2^k)_{ij} \end{pmatrix}, \quad i = 1, \dots, n_1, \quad j = 1, \dots, n_2, \quad (12)$$

where for $p, q = 1, 2$, the entries of the matrices $S^{pq} \in \mathbb{R}^{n_1 \times n_2}$ are given by

$$\begin{aligned} S_{ij}^{11} &= 1 - d_u \Delta t (\lambda_i^{(1)} + \lambda_j^{(2)}), & S_{ij}^{12} &= -d_{vu} \Delta t (\lambda_i^{(1)} + \lambda_j^{(2)}), \\ S_{ij}^{22} &= 1 - d_v \Delta t (\lambda_i^{(1)} + \lambda_j^{(2)}), & S_{ij}^{21} &= -d_{uv} \Delta t (\lambda_i^{(1)} + \lambda_j^{(2)}). \end{aligned}$$

The solution of the 2×2 linear system (12) for fixed i and j can be written as

$$\begin{pmatrix} \hat{U}_{ij}^k \\ \hat{V}_{ij}^k \end{pmatrix} = \begin{pmatrix} S_{ij}^{11} & S_{ij}^{12} \\ S_{ij}^{21} & S_{ij}^{22} \end{pmatrix}^{-1} \begin{pmatrix} (Q_1^k)_{ij} \\ (Q_2^k)_{ij} \end{pmatrix},$$

where for the determinant $|S_{ij}| = S_{ij}^{11} S_{ij}^{22} - S_{ij}^{12} S_{ij}^{21}$, the 2×2 inverse matrix can be easily calculated as

$$\begin{pmatrix} S_{ij}^{11} & S_{ij}^{12} \\ S_{ij}^{21} & S_{ij}^{22} \end{pmatrix}^{-1} = \frac{1}{|S_{ij}|} \begin{pmatrix} S_{ij}^{22} & -S_{ij}^{12} \\ -S_{ij}^{21} & S_{ij}^{11} \end{pmatrix},$$

Finally, by introducing the matrices $L^{pq} \in \mathbb{R}^{n_1 \times n_2}$, $p, q = 1, 2$, with the entries

$$L_{ij}^{11} = \frac{S_{ij}^{22}}{|S_{ij}|}, \quad L_{ij}^{12} = \frac{-S_{ij}^{12}}{|S_{ij}|}, \quad L_{ij}^{21} = \frac{-S_{ij}^{21}}{|S_{ij}|}, \quad L_{ij}^{22} = \frac{S_{ij}^{11}}{|S_{ij}|}, \quad (13)$$

the solution of the 2×2 linear system (12) are given by

$$\hat{U}_{ij}^k = L_{ij}^{11} (Q_1^k)_{ij} + L_{ij}^{12} (Q_2^k)_{ij}, \quad \hat{V}_{ij}^k = L_{ij}^{21} (Q_1^k)_{ij} + L_{ij}^{22} (Q_2^k)_{ij}.$$

The unknown solution matrices U^{k+1} and V^{k+1} can then be recovered by projecting back as $U^{k+1} = X \hat{U}^k Y^{-1}$ and $V^{k+1} = X \hat{V}^k Y^{-1}$, which can be written in terms of the matrices L^{pq} , Q_1^k and Q_2^k as

$$\begin{aligned} U^{k+1} &= X(L^{11} \odot Q_1^k + L^{12} \odot Q_2^k)Y^{-1}, \\ V^{k+1} &= X(L^{21} \odot Q_1^k + L^{22} \odot Q_2^k)Y^{-1}, \end{aligned}$$

where \odot denotes the Hadamard (element by element) product. The solution process to compute the full discrete solution matrices U^{k+1} and V^{k+1} by using IMEX Euler method applied to the semi-discrete linear matrix differential equation (7), and by utilizing the eigenvalue decompositions $T_1 = X \Lambda^{(1)} X^{-1}$ and $T_2^T = Y \Lambda^{(2)} Y^{-1}$ is given in Algorithm 1.

Algorithm 1 Solution process on a single time step

- 1: **Input:** Known solution matrices U^k and V^k , eigenvectors X and Y , eigenvalues $\Lambda^{(1)}$ and $\Lambda^{(2)}$
- 2: **Output:** Unknown solution matrices U^{k+1} and V^{k+1}
- 3: Compute the matrices Q_1^k and Q_2^k from (11)
- 4: Compute the matrices L^{11} , L^{12} , L^{21} and L^{22} from (13)
- 5: Compute the solution matrices U^{k+1} and V^{k+1} as

$$\begin{aligned} U^{k+1} &= X(L^{11} \odot Q_1^k + L^{12} \odot Q_2^k)Y^{-1} \\ V^{k+1} &= X(L^{21} \odot Q_1^k + L^{22} \odot Q_2^k)Y^{-1} \end{aligned}$$

The overall computational cost of solving two and three dimensional cross-diffusion systems is drastically reduced using the matrix/tensor formulation with the explicit-implicit time integration and using spectral decomposition. The computation of the spectral decomposition of the matrices T_1 and T_2 are performed once at the beginning of the integration.

2.4 Full discretization on three-dimensional domains

We consider a discrete mesh on a rectangular prism $\Omega = [0, \ell_x] \times [0, \ell_y] \times [0, \ell_z] \subset \mathbb{R}^3$ with the mesh sizes $h_x = \ell_x / (n_1 - 1)$, $h_y = \ell_y / (n_2 - 1)$ and $h_z = \ell_z / (n_3 - 1)$, and with the grid nodes $\mathbf{x}_{ijl} = (x_i, y_j, z_l)$, where $x_i = (i-1)h_x$, $y_j = (j-1)h_y$ and $z_l = (l-1)h_z$, $i = 1, \dots, n_1$, $j = 1, \dots, n_2$, $l = 1, \dots, n_3$.

The matrix oriented approach in [12] can be extended to the cross-diffusion systems (1) for the three-dimensional case following [10]. At each time step $t \in [0, t_f]$, let $U_{i,j}^l(t) = u(\mathbf{x}_{ijl}, t)$ and $V_{i,j}^l(t) = v(\mathbf{x}_{ijl}, t)$ denote the approximate

semi-discrete solutions at the grid nodes \mathbf{x}_{ijl} . Then, we introduce tall matrix functions $\mathbf{U}(t), \mathbf{V}(t) : [0, t_f] \mapsto \mathbb{R}^{(n_1 n_3) \times n_2}$ defined as

$$\mathbf{U}(t) = \begin{bmatrix} U^1(t) \\ U^2(t) \\ \vdots \\ U^{n_3}(t) \end{bmatrix}, \quad \mathbf{V}(t) = \begin{bmatrix} V^1(t) \\ V^2(t) \\ \vdots \\ V^{n_3}(t) \end{bmatrix}.$$

With this notation and an appropriate ordering of the nodes by the $\text{vec}(\cdot)$ operation introduced before, the terms in the system (4) including the matrix $A \in \mathbb{R}^{(n_1 n_2 n_3) \times (n_1 n_2 n_3)}$ and the solution vectors $\mathbf{u}, \mathbf{v} : [0, t_f] \mapsto \mathbb{R}^{(n_1 n_2 n_3)}$ can be written as

$$\begin{aligned} A\mathbf{u} &= \text{vec}((I_3 \otimes T_1)\mathbf{U} + \mathbf{U}T_2^T + (T_3 \otimes I_1)\mathbf{U}) \\ A\mathbf{v} &= \text{vec}((I_3 \otimes T_1)\mathbf{V} + \mathbf{V}T_2^T + (T_3 \otimes I_1)\mathbf{V}) \end{aligned} \quad (14)$$

At the discrete times $t_k = k\Delta t$, $k = 0, \dots, n_t$, let $\mathbf{U}^k = \mathbf{U}(t_k) \in \mathbb{R}^{(n_1 n_3) \times n_2}$ and $\mathbf{V}^k = \mathbf{V}(t_k) \in \mathbb{R}^{(n_1 n_3) \times n_2}$ denote the full discrete solution matrices at the time t_k . Then, using the identity (14), application of the IMEX Euler method, similar to the two-dimensional case, yields the following Sylvester equation as a matrix differential equation

$$\begin{aligned} (I_{13} - d_u \Delta t \hat{T})\mathbf{U}^{k+1} - d_u \Delta t \mathbf{U}^{k+1} T_2^T - d_{vu} \Delta t \hat{T} \mathbf{V}^{k+1} - d_{vu} \Delta t \mathbf{V}^{k+1} T_2^T &= \mathbf{U}^k + \Delta t \mathbf{F}(\mathbf{U}^k, \mathbf{V}^k), \\ (I_{13} - d_v \Delta t \hat{T})\mathbf{V}^{k+1} - d_v \Delta t \mathbf{V}^{k+1} T_2^T - d_{uv} \Delta t \hat{T} \mathbf{U}^{k+1} - d_{uv} \Delta t \mathbf{U}^{k+1} T_2^T &= \mathbf{V}^k + \Delta t \mathbf{G}(\mathbf{U}^k, \mathbf{V}^k), \end{aligned} \quad (15)$$

where $\hat{T} := T_1 \oplus T_3 = (I_3 \otimes T_1 + T_3 \otimes I_1) \in \mathbb{R}^{(n_1 n_3) \times (n_1 n_3)}$ with \oplus denoting the Kronecker sum, and I_{13} is the identity matrix of size $(n_1 n_3)$. The above Sylvester equation can be solved similar to the two-dimensional case. Here, it needs only the use of the eigenvalue decomposition of the matrix $\hat{T} = \hat{X} \hat{\Lambda} \hat{X}^{-1}$ in place of the eigenvalue decomposition of the matrix T_1 . However, the square matrix \hat{T} is of dimension $(n_1 n_3)$ which makes inefficient the computation of the eigenvalue decomposition of \hat{T} . Instead, we use the eigenvalue decomposition of the matrices T_1 , T_2^T and T_3 of smaller size, and we use the properties of Kronecker sum. Let the eigenvalue decompositions $T_1 = X \Lambda^{(1)} X^{-1}$, $T_2^T = Y \Lambda^{(2)} Y^{-1}$ and $T_3 = Z \Lambda^{(3)} Z^{-1}$ are given, with the matrices $X \in \mathbb{R}^{n_1 \times n_1}$, $Y \in \mathbb{R}^{n_2 \times n_2}$ and $Z \in \mathbb{R}^{n_3 \times n_3}$ of nonsingular vectors and the diagonal matrices $\Lambda^{(1)} = \text{diag}(\lambda_1^{(1)}, \dots, \lambda_{n_1}^{(1)})$, $\Lambda^{(2)} = \text{diag}(\lambda_1^{(2)}, \dots, \lambda_{n_2}^{(2)})$ and $\Lambda^{(3)} = \text{diag}(\lambda_1^{(3)}, \dots, \lambda_{n_3}^{(3)})$ of the eigenvalues $\lambda_i^{(1)}$, $\lambda_j^{(2)}$ and $\lambda_l^{(3)}$, $i = 1, \dots, n_1$, $j = 1, \dots, n_2$, $l = 1, \dots, n_3$. Then, by the use of the properties of the Kronecker sum, the eigenvalue decomposition of the matrix \hat{T} with the nonsingular vector \hat{X} and the diagonal matrix of the eigenvalues $\hat{\Lambda} = \text{diag}(\hat{\lambda}_1, \dots, \hat{\lambda}_{n_1 n_3})$ are given by

$$\hat{X} = (Z \otimes I_1)(I_3 \otimes X), \quad \hat{\Lambda} = (\Lambda^{(3)} \otimes I_1) + (I_3 \otimes \Lambda^{(1)}).$$

Similar to the two-dimensional case, the Sylvester equation (15) can be efficiently solved through multiplying both the equations in (15) from left by \hat{X}^{-1} and from right by Y , substituting $\hat{T} = \hat{X} \hat{\Lambda} \hat{X}^{-1}$ and $T_2^T = Y \Lambda^{(2)} Y^{-1}$, and setting $\hat{\mathbf{U}}^k = \hat{X}^{-1} \mathbf{U}^k Y$ and $\hat{\mathbf{V}}^k = \hat{X}^{-1} \mathbf{V}^k Y$, yielding the system

$$\begin{aligned} (I_{13} - d_u \Delta t \hat{\Lambda})\hat{\mathbf{U}}^{k+1} - d_u \Delta t \hat{\mathbf{U}}^{k+1} \Lambda^{(2)} - d_{vu} \Delta t \hat{\Lambda} \hat{\mathbf{V}}^{k+1} - d_{vu} \Delta t \hat{\mathbf{V}}^{k+1} \Lambda^{(2)} &= \mathbf{Q}_1^k, \\ (I_{13} - d_v \Delta t \hat{\Lambda})\hat{\mathbf{V}}^{k+1} - d_v \Delta t \hat{\mathbf{V}}^{k+1} \Lambda^{(2)} - d_{uv} \Delta t \hat{\Lambda} \hat{\mathbf{U}}^{k+1} - d_{uv} \Delta t \hat{\mathbf{U}}^{k+1} \Lambda^{(2)} &= \mathbf{Q}_2^k, \end{aligned} \quad (16)$$

where all the coefficient matrices on the left hand sides are again diagonal matrices, and the right hand side matrices are given by

$$\begin{aligned} \mathbf{Q}_1^k &= \hat{X}^{-1}(\mathbf{U}^k + \Delta t \mathbf{F}(\mathbf{U}^k, \mathbf{V}^k))Y, \\ \mathbf{Q}_2^k &= \hat{X}^{-1}(\mathbf{V}^k + \Delta t \mathbf{G}(\mathbf{U}^k, \mathbf{V}^k))Y. \end{aligned}$$

3 Nonintrusive reduced-order model

In this section, we consider the following parametrized form of the vectorial cross diffusion system (4)

$$\begin{aligned} \dot{\mathbf{u}}^\theta &= d_u A \mathbf{u}^\theta + d_{vu} A \mathbf{v}^\theta + \mathbf{f}(\mathbf{u}^\theta, \mathbf{v}^\theta; \theta), \quad \mathbf{u}^\theta(0) = \mathbf{u}_0^\theta, \\ \dot{\mathbf{v}}^\theta &= d_{uv} A \mathbf{u}^\theta + d_v A \mathbf{v}^\theta + \mathbf{g}(\mathbf{u}^\theta, \mathbf{v}^\theta; \theta), \quad \mathbf{v}^\theta(0) = \mathbf{v}_0^\theta, \end{aligned} \quad (17)$$

where the superscript $\theta \in \mathbf{P}$ indicates the parameter dependency of the solutions, and \mathbf{P} is a set of admissible values of the parameter θ which may stand for either parameter in the system. In most cases, the system (17) needs to be solved several times by the value of parameter θ differs. In this paper, by using a finite training set $\mathbf{P}_T = \{\theta_1, \dots, \theta_{n_p}\} \subset \mathbf{P}$,

we aim to construct a nonintrusive ROM to the system (17) in order to cheaply obtain approximate solutions for a given parameter value $\theta \in \mathbf{P}$ not necessarily from the training set \mathbf{P}_T , i.e., $\theta \notin \mathbf{P}_T$.

Reduced-order modelling methodology commonly relies on a data obtained from either an experiment or solutions of a discrete system like (17), which is named as snapshot data. In our case, solving the parametrized cross diffusion system (17) through either the matrix system (9) for a two-dimensional domain ($d = 2$) or the matrix system (15) for a three-dimensional domain ($d = 3$), and with a suitable arrangement of the dimensions, we can obtain a set of snapshots $\{\mathcal{U}^\theta(t_k)\}_{k=1}^{n_t}$ and $\{\mathcal{V}^\theta(t_k)\}_{k=1}^{n_t}$ in the form of an order- d tensor (multidimensional array) with $\mathcal{U}^\theta(t_k) \in \mathbb{R}^{n_1 \times \dots \times n_d}$ and $\mathcal{V}^\theta(t_k) \in \mathbb{R}^{n_1 \times \dots \times n_d}$. Here, each dimension of the tensors $\mathcal{U}^\theta(t_k)$ and $\mathcal{V}^\theta(t_k)$ corresponds to one of the respective spatial directions, for instance, with $d = 3$, $\mathcal{U}_{i_1 i_2 i_3}^\theta(t_k) = u^\theta(\mathbf{x}_{i_1 i_2 i_3}, t_k)$, $i_s = 1, \dots, n_s$, $s = 1, 2, 3$. Then, we form the following order- $(d + 1)$ tensors of snapshot data related to a given parameter θ

$$\begin{aligned} \mathcal{X}^{\theta,u} &\in \mathbb{R}^{n_1 \times \dots \times n_d \times n_{d+1}}, & \mathcal{X}_{i_1 \dots i_d i_{d+1}}^{\theta,u} &= \mathcal{U}_{i_1 \dots i_d}^\theta(t_{i_{d+1}}), \\ \mathcal{X}^{\theta,v} &\in \mathbb{R}^{n_1 \times \dots \times n_d \times n_{d+1}}, & \mathcal{X}_{i_1 \dots i_d i_{d+1}}^{\theta,v} &= \mathcal{V}_{i_1 \dots i_d}^\theta(t_{i_{d+1}}), \end{aligned} \quad (18)$$

where for easy notation we set the size of the final dimension related to the time as $n_{d+1} := n_t$.

The standard POD approach to construct the reduced basis solutions for many training parameter values is costly. The two-level POD, known also as nested POD, is often used in ROM applications for parametrized PDEs to reduce the computational cost of constructing the spatial and temporal basis functions [19, 20]. Usually, a snapshot data which is in the form of columns consisting of the solutions in vector form, is used in this ROM methodology. In the first level, a set of POD basis are computed for the snapshot data related to each parameter $\theta_i \in \mathbf{P}_T$. Then, in the second level, a global POD basis is constructed by applying SVD to the set of POD basis computed in the first level [19, 20]. Finally, the space-time coefficients are determined using RBF in a nonintrusive way without resorting to Galerkin projection.

For the snapshot data in the form of an order-2 tensor, i.e., a matrix, the POD basis in the first level are computed by applying SVD or eigenvalue decomposition to the snapshot matrix. However, in our case, each snapshot data given in (18) is an order- $(d + 1)$ tensor with $d = 2, 3$. Here, in the first level of nested POD, we compute the POD modes using HOSVD of the snapshot tensors $\mathcal{X}^{\theta,u}$ and $\mathcal{X}^{\theta,v}$. The HOSVD is a favorite algorithm for computing low-rank approximation of the Tucker decomposition of a tensor [27]. In the following, we will describe HOSVD to compute the POD modes of the snapshot tensor $\mathcal{X}^{\theta,u}$, the POD modes of the snapshot tensor $\mathcal{X}^{\theta,v}$ can be computed similarly. Like any multidimensional array, the order- $(d + 1)$ snapshot tensor $\mathcal{X}^{\theta,u} \in \mathbb{R}^{n_1 \times \dots \times n_d \times n_{d+1}}$ admits the following Tucker decomposition [23, 27, 28]

$$\mathcal{X}_{i_1 \dots i_d i_{d+1}}^{\theta,u} = \sum_{j_1=1}^{n_1} \dots \sum_{j_d=1}^{n_d} \sum_{j_{d+1}=1}^{n_{d+1}} \mathcal{S}_{j_1 \dots j_d j_{d+1}}^{\theta,u} \Phi_{i_1 j_1}^{(1),\theta,u} \dots \Phi_{i_d j_d}^{(d),\theta,u} \Phi_{i_{d+1} j_{d+1}}^{(d+1),\theta,u},$$

or in a suitable compact form

$$\mathcal{X}^{\theta,u} = \left(\Phi^{(1),\theta,u}, \dots, \Phi^{(d),\theta,u}, \Phi^{(d+1),\theta,u} \right) \cdot \mathcal{S}^{\theta,u}, \quad (19)$$

where the order- $(d + 1)$ tensor $\mathcal{S}^{\theta,u} \in \mathbb{R}^{n_1 \times \dots \times n_d \times n_{d+1}}$ is called the core tensor, and each orthonormal matrix $\Phi^{(j),\theta,u} \in \mathbb{R}^{n_j \times n_j}$, $j = 1, \dots, (d + 1)$, is called a factor matrix. In other words, a tensor can be decomposed into a core tensor that is multiplied by a matrix along each mode, which are orthonormal and can be viewed as the principal components of each modes. The HOSVD aims firstly to compute the factor matrices $\Phi^{(j),\theta,u}$, $j = 1, \dots, (d + 1)$. This process is done by applying SVD to the mode- j unfolding (matricization) $\mathcal{X}_{(j)}^{\theta,u}$ of the tensor $\mathcal{X}^{\theta,u}$, where a mode- j unfolding $\mathcal{X}_{(j)}^{\theta,u}$ is a matrix of size $n_j \times \prod_{i \neq j} n_i$, and its columns are mode- j fibers of the tensor $\mathcal{X}^{\theta,u}$ [23, 27, 28]. Then, the j th factor matrix is given by the left singular vectors of the mode- j unfolding of the tensor $\mathcal{X}^{\theta,u}$

$$\mathcal{X}_{(j)}^{\theta,u} = \Phi^{(j),\theta,u} \Sigma^{(j),\theta,u} \left(\psi^{(j),\theta,u} \right)^T, \quad j = 1, \dots, (d + 1),$$

where the diagonal matrix $\Sigma^{(j),\theta,u} \in \mathbb{R}^{n_j \times n_j}$ includes on its diagonal elements the singular values $\sigma_i \left(\mathcal{X}_{(j)}^{\theta,u} \right) \geq 0$ of the mode- j unfolding $\mathcal{X}_{(j)}^{\theta,u}$, $i = 1, \dots, n_j$. After computation of the factor matrices, the core tensor can be calculated as

$$\mathcal{S}^{\theta,u} = \left(\left(\Phi^{(1),\theta,u} \right)^T, \dots, \left(\Phi^{(d),\theta,u} \right)^T, \left(\Phi^{(d+1),\theta,u} \right)^T \right) \cdot \mathcal{X}^{\theta,u}.$$

By the use of HOSVD, the factor matrices take place of the POD modes required in the first level of the nested POD, each of which corresponds to one of the either space direction or temporal dimension. In addition, the space-time

coefficients are contained in the core tensor $\mathcal{S}^{\theta,u}$, therefore there is no need to determine them through the solution of a separate system as in [19, 20].

On the other hand, the decomposition (19) does not provide a low-rank approximation yet, it requires $\prod_{i=1}^{d+1} n_i + \sum_{i=1}^{d+1} n_i^2$ numbers to be stored. The HOSVD can be employed to construct a low multilinear rank approximation to a tensor, where it provides a compressed representation in the Tucker decomposition. One approach is the truncated HOSVD (T-HOSVD) which was first introduced in [22]. The T-HOSVD algorithm aims to compute each factor matrix separately, and it relies on truncating each mode- j unfolding $\mathcal{X}_{(j)}^{\theta,u}$ of the snapshot tensor $\mathcal{X}^{\theta,u}$ according to a given truncation criteria or an a priori given target ranks for each dimension j . Let for some target rank $\mathbf{r}^{\theta,u} = (r_1^{\theta,u}, \dots, r_{d+1}^{\theta,u})$ with $r_j^{\theta,u} < n_j$ for each $j = 1, \dots, (d+1)$, the truncated SVD of the mode- j unfolding $\mathcal{X}_{(j)}^{\theta,u}$ is given by

$$\mathcal{X}_{(j)}^{\theta,u} = \Phi^{(j),\theta,u} \Sigma^{(j),\theta,u} \left(\psi^{(j),\theta,u} \right)^T = \left[\bar{\Phi}^{(j),\theta,u} \quad \tilde{\Phi}^{(j),\theta,u} \right] \begin{bmatrix} \bar{\Sigma}^{(j),\theta,u} & \\ & \tilde{\Sigma}^{(j),\theta,u} \end{bmatrix} \begin{bmatrix} \left(\bar{\psi}^{(j),\theta,u} \right)^T \\ \left(\tilde{\psi}^{(j),\theta,u} \right)^T \end{bmatrix}, \quad (20)$$

where $\bar{\Phi}^{(j),\theta,u} \in \mathbb{R}^{n_j \times r_j^{\theta,u}}$ contains the first $r_j^{\theta,u}$ left singular vectors from $\Phi^{(j),\theta,u}$, retained singular values are contained in $\bar{\Sigma}^{(j),\theta,u} \in \mathbb{R}^{r_j^{\theta,u} \times r_j^{\theta,u}}$, and $\tilde{\Sigma}^{(j),\theta,u}$ contains the truncated singular values. Using the truncated factor matrices $\bar{\Phi}^{(j),\theta,u}$, we can calculate the truncated (reduced) core tensor $\bar{\mathcal{S}}^{\theta,u} \in \mathbb{R}^{r_1^{\theta,u} \times \dots \times r_d^{\theta,u} \times r_{d+1}^{\theta,u}}$ using the formula

$$\bar{\mathcal{S}}^{\theta,u} = \left(\left(\bar{\Phi}^{(1),\theta,u} \right)^T, \dots, \left(\bar{\Phi}^{(d),\theta,u} \right)^T, \left(\bar{\Phi}^{(d+1),\theta,u} \right)^T \right) \cdot \mathcal{X}^{\theta,u}.$$

Then, a rank- $(r_1^{\theta,u}, \dots, r_d^{\theta,u}, r_{d+1}^{\theta,u})$ approximation $\bar{\mathcal{X}}^{\theta,u} \in \mathbb{R}^{n_1 \times \dots \times n_d \times n_{d+1}}$ to the snapshot tensor $\mathcal{X}^{\theta,u} \in \mathbb{R}^{n_1 \times \dots \times n_d \times n_{d+1}}$ can be obtained as

$$\mathcal{X}^{\theta,u} \approx \bar{\mathcal{X}}^{\theta,u} = \left(\bar{\Phi}^{(1),\theta,u}, \dots, \bar{\Phi}^{(d),\theta,u}, \bar{\Phi}^{(d+1),\theta,u} \right) \cdot \bar{\mathcal{S}}^{\theta,u}, \quad (21)$$

where it stores only $\prod_{i=1}^{d+1} r_i^{\theta,u} + \sum_{i=1}^{d+1} n_i r_i^{\theta,u}$ numbers. The quantity that to what extent the memory saving is obtained, can be visualized by the following compression factor [29]

$$C_F = \frac{\prod_{i=1}^{d+1} n_i + \sum_{i=1}^{d+1} n_i^2}{\prod_{i=1}^{d+1} r_i^{\theta,u} + \sum_{i=1}^{d+1} n_i r_i^{\theta,u}}, \quad (22)$$

which gives the saved memory in percentage by the formula $100(1 - 1/C_F)$. The larger the compression factor C_F the much more the memory is saved.

Although, T-HOSVD provides a low multilinear rank approximation, the SVD computations of the unfoldings may be expensive, since the same full rank tensor is used to obtain each unfolding. Another approach to construct a low multilinear rank approximation is the sequentially truncated HOSVD (ST-HOSVD) [23, 28], which is a variation of the usual T-HOSVD. In the ST-HOSVD, instead of throwing away most of the work performed by each SVD, SVD is performed sequentially on a reduced tensor along all dimensions. Starting from the initial core tensor $\bar{\mathcal{S}}^{\theta,u,(0)} := \mathcal{X}^{\theta,u}$, ST-HOSVD computes a sequence of core tensors $\bar{\mathcal{S}}^{\theta,u,(j)}$ to reach the reduced core tensor $\bar{\mathcal{S}}^{\theta,u} = \bar{\mathcal{S}}^{\theta,u,(d+1)}$ following an ordering $\mathbf{p} = (p_1, \dots, p_{d+1})$ which is a permutation of the index set $(1, 2, \dots, d+1)$. In the j th stage, the truncated factor matrix $\bar{\Phi}^{(p_j),\theta,u}$ of the mode- p_j unfolding of the core tensor $\bar{\mathcal{S}}^{\theta,u,(j-1)}$ is computed, and the new core tensor $\bar{\mathcal{S}}^{\theta,u,(j)}$ is calculated by projecting the previous one onto the subspace spanned by the columns of the computed factor matrix $\bar{\Phi}^{(p_j),\theta,u}$. The ST-HOSVD algorithm is given in Algorithm 2 [28].

Unlike T-HOSVD, the process in the ST-HOSVD is sequential, therefore the order in which the modes are processed affects the accuracy of the approximation and the speed of the process. In [23], a heuristic is proposed that attempts to minimize the number of operations required to compute the dominant subspace. Processing first the dimension with the lowest size may even reduce the rank of the remaining terms, i.e., $n_{p_1} \leq n_{p_2} \leq \dots \leq n_{p_{d+1}}$. In this way, more energy is forced into fewer modes. Computing the T-HOSVD can be more expensive than the ST-HOSVD, while the ST-HOSVD requires fewer floating point operations to compute the approximation. Although, T-HOSVD and ST-HOSVD approximations may differ in accuracy for an ordering $\mathbf{p} \neq (1, \dots, d+1)$, both T-HOSVD and ST-HOSVD approximations satisfy the same error bounds [23]

$$\min_{1 \leq j \leq d+1} \|\tilde{\Sigma}^{(j),\theta,u}\|_F^2 \leq \|\mathcal{X}^{\theta,u} - \bar{\mathcal{X}}^{\theta,u}\|_F^2 \leq \sum_{j=1}^{d+1} \|\tilde{\Sigma}^{(j),\theta,u}\|_F^2, \quad (23)$$

Algorithm 2 ST-HOSVD for tensor data of u component

-
- 1: **Input:** Snapshot tensor $\mathcal{X}^{\theta,u}$, processing order $\mathbf{p} = (p_1, \dots, p_{d+1})$, target ranks $\mathbf{r}^{\theta,u} = (r_1^{\theta,u}, \dots, r_{d+1}^{\theta,u})$
 - 2: **Output:** Truncated factor matrices $\{\bar{\Phi}^{(1),\theta,u}, \dots, \bar{\Phi}^{(d+1),\theta,u}\}$
 - 3: Set $\bar{\mathcal{S}}^{\theta,u,(0)} = \mathcal{X}^{\theta,u}$
 - 4: **for** $j = 1$ to $d + 1$ **do**
 - 5: Obtain mode- p_j unfolding $\bar{\mathcal{S}}_{(p_j)}^{\theta,u,(j-1)}$
 - 6: Apply truncated SVD to the unfolding $\bar{\mathcal{S}}_{(p_j)}^{\theta,u,(j-1)}$ for target rank $r_{p_j}^{\theta,u}$
 - 7: Get the factor matrix $\bar{\Phi}_{(p_j),\theta,u}^{(j)}$
 - 8: Update the unfolding $\bar{\mathcal{S}}_{(p_j)}^{\theta,u,(j-1)} \leftarrow \bar{\Sigma}_{(p_j),\theta,u}^{(j)} (\bar{\psi}_{(p_j),\theta,u}^{(j)})^T$
 - 9: Obtain updated core tensor $\bar{\mathcal{S}}_{(p_j)}^{\theta,u,(j)} \leftarrow \bar{\mathcal{S}}_{(p_j)}^{\theta,u,(j-1)}$ in tensor form
 - 10: **end for**
-

where $\|\cdot\|_F$ is the Frobenius norm, and $\tilde{\Sigma}^{(j),\theta,u}$ contains the truncated singular values given in (20).

In order to construct the nonintrusive ROM through the nested POD, we first form the set of snapshot tensors $\{\mathcal{X}^{\theta_i,u}\}_{i=1}^{n_p}$ and $\{\mathcal{X}^{\theta_i,v}\}_{i=1}^{n_p}$ from the solutions of the parametrized cross diffusion system related to each sample parameter $\theta_i \in \mathbf{P}_T$. Then, in the first level of the nested POD, we apply ST-HOSVD to the snapshot tensors $\mathcal{X}^{\theta_i,u}$ and $\mathcal{X}^{\theta_i,v}$, and we collect related to each sample parameter $\theta_i \in \mathbf{P}_T$, the truncated factor matrices $\{\bar{\Phi}^{(j),\theta_i,u}\}_{i=1}^{n_p}$ and $\{\bar{\Phi}^{(j),\theta_i,v}\}_{i=1}^{n_p}$ with the target ranks $\mathbf{r}^{\theta_i,u} = (r_1^{\theta_i,u}, \dots, r_{d+1}^{\theta_i,u})$ and $\mathbf{r}^{\theta_i,v} = (r_1^{\theta_i,v}, \dots, r_{d+1}^{\theta_i,v})$, respectively, $j = 1, \dots, (d+1)$. Then, in the second level of the nested POD, we compute the truncated global factor matrices $\hat{\Phi}^{(j),u} \in \mathbb{R}^{n_j \times \hat{r}_j^u}$ and $\hat{\Phi}^{(j),v} \in \mathbb{R}^{n_j \times \hat{r}_j^v}$, $j = 1, \dots, (d+1)$, as the truncated left singular vectors obtained by the application of the truncated SVD to the collections $\bar{\Phi}^{(j),u}$ and $\bar{\Phi}^{(j),v}$ of the factor matrices defined by

$$\begin{aligned} \bar{\Phi}^{(j),u} &= [\bar{\Phi}^{(j),\theta_1,u} \bar{\Phi}^{(j),\theta_2,u} \dots \bar{\Phi}^{(j),\theta_{n_p},u}] \in \mathbb{R}^{n_j \times \bar{r}_j^u}, \\ \bar{\Phi}^{(j),v} &= [\bar{\Phi}^{(j),\theta_1,v} \bar{\Phi}^{(j),\theta_2,v} \dots \bar{\Phi}^{(j),\theta_{n_p},v}] \in \mathbb{R}^{n_j \times \bar{r}_j^v}, \end{aligned}$$

where the numbers $\bar{r}_j^u := r_j^{\theta_1,u} + \dots + r_j^{\theta_{n_p},u}$ and $\bar{r}_j^v := r_j^{\theta_1,v} + \dots + r_j^{\theta_{n_p},v}$ denote the column size of the collection of the factor matrices, which are the sum of the target ranks $r_j^{\theta_i,u}$ and $r_j^{\theta_i,v}$ of each unfolding $\mathcal{X}^{\theta_i,u}$ and $\mathcal{X}^{\theta_i,v}$, respectively, $i = 1, \dots, n_p$, $j = 1, \dots, (d+1)$. In addition, the numbers $\hat{r}_j^u < \bar{r}_j^u$ and $\hat{r}_j^v < \bar{r}_j^v$ are the target ranks of the collections $\bar{\Phi}^{(j),u}$ and $\bar{\Phi}^{(j),v}$ of the factor matrices, respectively. Note that the truncated global factor matrices $\hat{\Phi}^{(j),u}$ and $\hat{\Phi}^{(j),v}$ are independent of the parameter θ , they rely on the sample parameter set \mathbf{P}_T . Once the truncated global factor matrices $\hat{\Phi}^{(j),u}$ and $\hat{\Phi}^{(j),v}$ are obtained, we compute for each sample parameter $\theta_i \in \mathbf{P}_T$, the core tensors $\hat{\mathcal{S}}^{\theta_i,u} \in \mathbb{R}^{\hat{r}_1^u \times \dots \times \hat{r}_d^u \times \hat{r}_{d+1}^u}$ and $\hat{\mathcal{S}}^{\theta_i,v} \in \mathbb{R}^{\hat{r}_1^v \times \dots \times \hat{r}_d^v \times \hat{r}_{d+1}^v}$ as

$$\begin{aligned} \hat{\mathcal{S}}^{\theta_i,u} &= \left(\left(\hat{\Phi}^{(1),u} \right)^T, \dots, \left(\hat{\Phi}^{(d),u} \right)^T, \left(\hat{\Phi}^{(d+1),u} \right)^T \right) \cdot \mathcal{X}^{\theta_i,u}, \\ \hat{\mathcal{S}}^{\theta_i,v} &= \left(\left(\hat{\Phi}^{(1),v} \right)^T, \dots, \left(\hat{\Phi}^{(d),v} \right)^T, \left(\hat{\Phi}^{(d+1),v} \right)^T \right) \cdot \mathcal{X}^{\theta_i,v}, \quad i = 1, \dots, n_p. \end{aligned}$$

Finally, the parameter dependent nonintrusive ROM solution tensors $\hat{\mathcal{X}}^u(\theta), \hat{\mathcal{X}}^v(\theta) \in \mathbb{R}^{n_1 \times \dots \times n_d \times n_{d+1}}$ for an arbitrary parameter $\theta \in \mathbf{P}$ can be efficiently obtained by the formulas

$$\begin{aligned} \mathcal{X}^{\theta,u} &\approx \hat{\mathcal{X}}^u(\theta) = \left(\hat{\Phi}^{(1),u}, \dots, \hat{\Phi}^{(d),u}, \hat{\Phi}^{(d+1),u} \right) \cdot \hat{\mathcal{S}}^u(\theta), \\ \mathcal{X}^{\theta,v} &\approx \hat{\mathcal{X}}^v(\theta) = \left(\hat{\Phi}^{(1),v}, \dots, \hat{\Phi}^{(d),v}, \hat{\Phi}^{(d+1),v} \right) \cdot \hat{\mathcal{S}}^v(\theta), \end{aligned} \quad (24)$$

where the parameter dependent global core tensors $\hat{\mathcal{S}}^u(\theta) : \mathbf{P} \mapsto \mathbb{R}^{\hat{r}_1^u \times \dots \times \hat{r}_d^u \times \hat{r}_{d+1}^u}$ and $\hat{\mathcal{S}}^v(\theta) : \mathbf{P} \mapsto \mathbb{R}^{\hat{r}_1^v \times \dots \times \hat{r}_d^v \times \hat{r}_{d+1}^v}$ stands for the data of undetermined coefficients, which can be easily determined by a variety of methods. Here, each

entry of the global core tensors are expanded using RBFs as follows

$$\begin{aligned}\widehat{\mathcal{S}}_{j_1 \dots j_d j_{d+1}}^u(\theta) &= \sum_{j=1}^{n_p} \gamma_{j_1 \dots j_d j_{d+1}}^{u,j} \Psi(\omega_j(\theta)), \\ \widehat{\mathcal{S}}_{j_1 \dots j_d j_{d+1}}^v(\theta) &= \sum_{j=1}^{n_p} \gamma_{j_1 \dots j_d j_{d+1}}^{v,j} \Psi(\omega_j(\theta)),\end{aligned}\tag{25}$$

where $\Psi(\omega)$ denote the radial basis kernel function with $\omega_j(\theta) = |\theta - \theta_j|$, and the scalars $\gamma_{j_1 \dots j_d j_{d+1}}^{u,j}$ and $\gamma_{j_1 \dots j_d j_{d+1}}^{v,j}$ are the coefficients to be determined. RBF is a real-valued function whose value depends on the distance from center point so that $\Psi(\omega) = \Psi(\|\omega\|)$ is a radial function. There exist well-known RBFs. Here, Gaussian RBF $\Psi(\omega) = e^{(-\omega^2/2\rho)}$ is used, where the parameter ρ , in our case, is given by $\rho = (\max_i \theta_i - \min_i \theta_i)/n_p$.

In order to compute the undetermined coefficients $\gamma_{j_1 \dots j_d j_{d+1}}^{u,j}$ and $\gamma_{j_1 \dots j_d j_{d+1}}^{v,j}$, we use the core tensors $\widehat{\mathcal{S}}^{\theta_i, u}$ and $\widehat{\mathcal{S}}^{\theta_i, v}$. Setting $\theta = \theta_i$ in (25) with the properties that $\widehat{\mathcal{S}}^u(\theta_i) = \widehat{\mathcal{S}}^{\theta_i, u}$ and $\widehat{\mathcal{S}}^v(\theta_i) = \widehat{\mathcal{S}}^{\theta_i, v}$, we obtain that

$$\begin{aligned}\widehat{\mathcal{S}}_{j_1 \dots j_d j_{d+1}}^u(\theta_i) &= \widehat{\mathcal{S}}_{j_1 \dots j_d j_{d+1}}^{\theta_i, u} = \sum_{j=1}^{n_p} \gamma_{j_1 \dots j_d j_{d+1}}^{u,j} \Psi(\omega_j(\theta_i)), \\ \widehat{\mathcal{S}}_{j_1 \dots j_d j_{d+1}}^v(\theta_i) &= \widehat{\mathcal{S}}_{j_1 \dots j_d j_{d+1}}^{\theta_i, v} = \sum_{j=1}^{n_p} \gamma_{j_1 \dots j_d j_{d+1}}^{v,j} \Psi(\omega_j(\theta_i)),\end{aligned}\quad i = 1, \dots, n_p,$$

which leads to the following linear systems of equations

$$\sum_{j=1}^{n_p} B_{ij} \gamma_{j_1 \dots j_d j_{d+1}}^{u,j} = \widehat{\mathcal{S}}_{j_1 \dots j_d j_{d+1}}^{\theta_i, u}, \quad \sum_{j=1}^{n_p} B_{ij} \gamma_{j_1 \dots j_d j_{d+1}}^{v,j} = \widehat{\mathcal{S}}_{j_1 \dots j_d j_{d+1}}^{\theta_i, v}, \quad i = 1, \dots, n_p,$$

where the entries of the symmetric interpolation matrix $B \in \mathbb{R}^{n_p \times n_p}$ is given by $B_{ij} = \Psi(\omega_j(\theta_i))$. In short, for a given new parameter value $\theta \in \mathcal{P}$, once the undetermined coefficients $\gamma_{j_1 \dots j_d j_{d+1}}^{u,j}$ and $\gamma_{j_1 \dots j_d j_{d+1}}^{v,j}$ in (25) are computed by the RBF interpolation, the nonintrusive ROM solution tensors $\widehat{\mathcal{X}}^u(\theta)$ and $\widehat{\mathcal{X}}^v(\theta)$ in (24) are calculated, by which the nonintrusive ROM approximations $\widehat{u}(\mathbf{x}, t; \theta) \approx u(\mathbf{x}, t; \theta)$ and $\widehat{v}(\mathbf{x}, t; \theta) \approx v(\mathbf{x}, t; \theta)$ can be cheaply obtained as

$$\widehat{u}(\mathbf{x}_{i_1 \dots i_d}, t_k; \theta) = \widehat{\mathcal{X}}_{i_1 \dots i_d}^u(\theta), \quad \widehat{v}(\mathbf{x}_{i_1 \dots i_d}, t_k; \theta) = \widehat{\mathcal{X}}_{i_1 \dots i_d}^v(\theta).$$

4 Numerical results

In this section we report about the numerical tests for the two-dimensional Schnakenberg (2) and three-dimensional Brusselator (3) cross-diffusion systems. All the simulations are performed on a machine with Intel Core™ i7 2.5 GHz 64 bit CPU, 8 GB RAM, Windows 10, using 64 bit MatLab R2014. For both problems, the initial conditions are taken as random periodic perturbation around the equilibrium solutions u_e and v_e

- Schnakenberg [5]:

$$u_0(\mathbf{x}) = u_e + \text{rand}(\mathbf{x})/100, \quad v_0(\mathbf{x}) = v_e + \text{rand}(\mathbf{x})/100,$$

with $(u_e, v_e) = (0.55, 0.9917)$,

- Brusselator [6]:

$$u_0(\mathbf{x}) = u_e + \text{rand}(\mathbf{x})/3, \quad v_0(\mathbf{x}) = v_e + \text{rand}(\mathbf{x})/10,$$

with $(u_e, v_e) = (6, 0.1667)$,

where $\text{rand}(\mathbf{x})$ is the MatLab's random function producing a multi-dimensional array of the same dimension as \mathbf{x} , with the entries are uniformly distributed random numbers between 0 and 1.

For a given parameter value $\theta \in \mathcal{P}$, the accuracy of the corresponding reduced approximations are measured using the time averaged relative errors

$$\begin{aligned}\|u - \widehat{u}\|_{\text{rel}} &= \frac{1}{n_t} \sum_{k=1}^{n_t} \frac{\|\mathcal{U}^\theta(t_k) - \widehat{\mathcal{U}}^\theta(t_k)\|_F}{\|\mathcal{U}^\theta(t_k)\|_F}, \\ \|v - \widehat{v}\|_{\text{rel}} &= \frac{1}{n_t} \sum_{k=1}^{n_t} \frac{\|\mathcal{V}^\theta(t_k) - \widehat{\mathcal{V}}^\theta(t_k)\|_F}{\|\mathcal{V}^\theta(t_k)\|_F},\end{aligned}\tag{26}$$

where $\mathcal{U}^\theta(t), \mathcal{V}^\theta(t) : [0, t_f] \mapsto \mathbb{R}^{n_1 \times \dots \times n_d}$ are the discrete FOM solutions while $\widehat{\mathcal{U}}^\theta(t), \widehat{\mathcal{V}}^\theta(t) : [0, t_f] \mapsto \mathbb{R}^{n_1 \times \dots \times n_d}$ are the discrete ROM approximations in the form of order- d tensor ($d = 2, 3$).

In both examples, in order to obtain the truncated factor matrices in the first level of nested POD, say $\bar{\Phi}^{(j),\theta,u}$, the target rank $r_j^{\theta,u}$ for each unfolding $\mathcal{X}_{(j)}^{\theta,u}$, $j = 1, \dots, (d+1)$, is determined compatibly with the error bound (23) in Frobenius norm, so that the following criteria is satisfied for a user given tolerance $\tau_1 > 0$

$$\frac{\sqrt{\sum_{i=r_j^{\theta,u}+1}^{n_j} \sigma_i(\mathcal{X}_{(j)}^{\theta,u})}}{\sqrt{\sum_{i=1}^{n_j} \sigma_i(\mathcal{X}_{(j)}^{\theta,u})}} < \tau_1. \quad (27)$$

On the other hand, in the second level of nested POD, in order obtain the truncated global factor matrices, say $\widehat{\Phi}^{(j),u}$, we apply the truncated SVD to the collection $\bar{\Phi}^{(j),u} = [\bar{\Phi}^{(j),\theta_1,u} \ \bar{\Phi}^{(j),\theta_2,u} \ \dots \ \bar{\Phi}^{(j),\theta_{n_p},u}] \in \mathbb{R}^{n_j \times \bar{r}_j^u}$ ($\bar{r}_j^u = r_j^{\theta_1,u} + \dots + r_j^{\theta_{n_p},u}$) of the truncated factor matrices with the target rank $\widehat{r}_j^u < \bar{r}_j^u$ which is determined so that the following energy criteria is satisfied for a user given tolerance $\tau_2 > 0$

$$\frac{\sum_{i=1}^{\widehat{r}_j^u} \sigma_i(\bar{\Phi}^{(j),u})}{\sum_{i=1}^{\bar{r}_j^u} \sigma_i(\bar{\Phi}^{(j),u})} \geq 1 - \tau_2, \quad (28)$$

where $\sigma_i(\bar{\Phi}^{(j),u}) \geq 0$ are the singular values of the collection $\bar{\Phi}^{(j),u}$ of the truncated factor matrices computed in the first level. In our simulations, we choose the user given tolerances scaling as $\tau_1 \sim 10^{-2}$ and $\tau_2 \sim 10^{-8}$.

4.1 Schnakenberg equation

We consider the two-dimensional Schnakenberg model [5] in the square domain $\Omega = [0, 0.5]^2 \subset \mathbb{R}^2$. We solve the problem through the matrix equation (10) with the mesh sizes $h_x = h_y = 0.005$ for the number of grid points $n_1 = n_2 = 101$. The final time is taken as $t_f = 5$ with the time step size $\Delta t = 0.001$, leading to the third dimension $n_3 = 5001$ of the snapshot tensors related to the time. For this problem, we fix the system parameters $d_u = d_v = d_{vu} = 1$, $\gamma = 200$, $\alpha = 0.25$, $\beta = 0.3$, and vary the cross-diffusion parameter $\theta := d_{uv}$ in the set of admissible values $P = [0.4, 0.8]$. As the finite training set of parameter θ , we take the values (including the boundary values) uniformly distributed on P with the increment 0.1, i.e., $P_T = \{0.4, 0.5, 0.6, 0.7, 0.8\}$ with the number of sample parameter values $n_p = 5$.

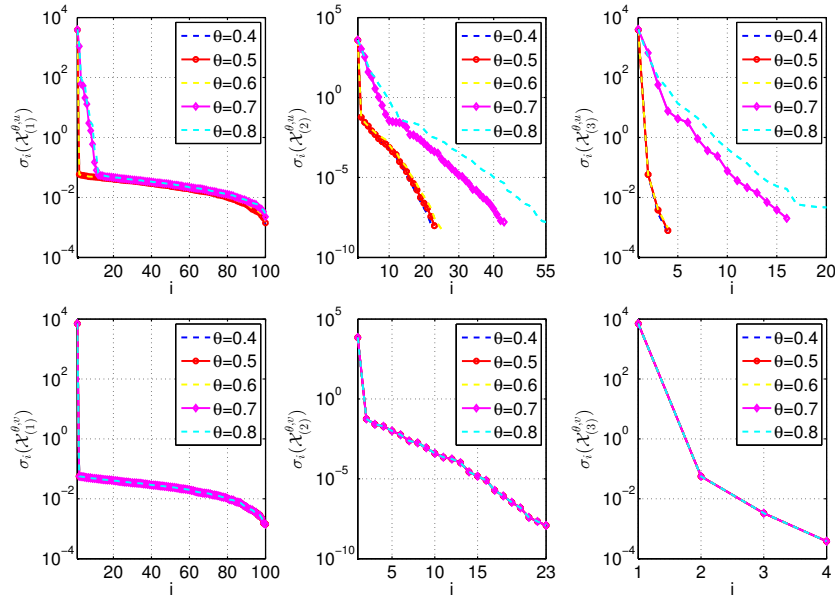


Figure 1: Schnakenberg model: Level I singular values of unfoldings $\mathcal{X}_{(j)}^{\theta,u}$ (top) and $\mathcal{X}_{(j)}^{\theta,v}$ (bottom)

In Figure 1, we give the decay of the singular values $\sigma_i(\mathcal{X}_{(j)}^{\theta,u})$ and $\sigma_i(\mathcal{X}_{(j)}^{\theta,v})$ of the unfoldings $\mathcal{X}_{(j)}^{\theta,u}$ and $\mathcal{X}_{(j)}^{\theta,v}$ of the order-3 snapshot tensors $\mathcal{X}^{\theta,u}$ and $\mathcal{X}^{\theta,v}$, respectively, $j = 1, 2, 3$, related to each sample parameter value $\theta \in \mathbf{P}_T$. According to the criteria (27), the computed target ranks $r_j^{\theta,u}$ and $r_j^{\theta,v}$ required by the HOSVD in the first level of nested POD are presented in Figure 2, which shows in accordance with the singular values in Figure 1 that enough energetic part of the unfoldings are recovered.

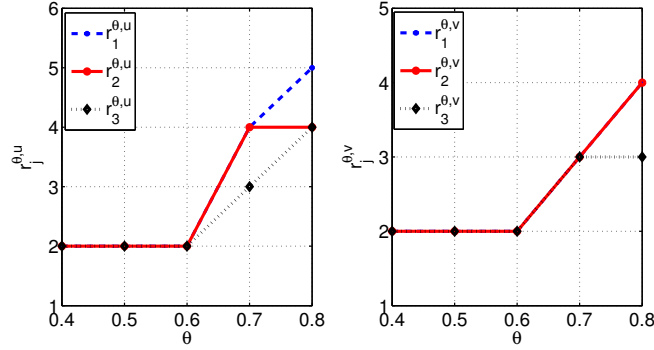


Figure 2: Schnakenberg model: Sample parameter values vs target ranks for unfoldings $\mathcal{X}_{(j)}^{\theta,u}$ (left) and $\mathcal{X}_{(j)}^{\theta,v}$ (right) at Level I

The FOM solutions $u(\mathbf{x}, t; \theta)$ and $v(\mathbf{x}, t; \theta)$ together with the nonintrusive ROM approximations $\hat{u}(\mathbf{x}, t; \theta)$ and $\hat{v}(\mathbf{x}, t; \theta)$ at the final time $t_f = 5$ for the parameter value $\theta = 0.65 \notin \mathbf{P}_T$ are given in Figure 3. We see from the figures that the same patterns are caught.

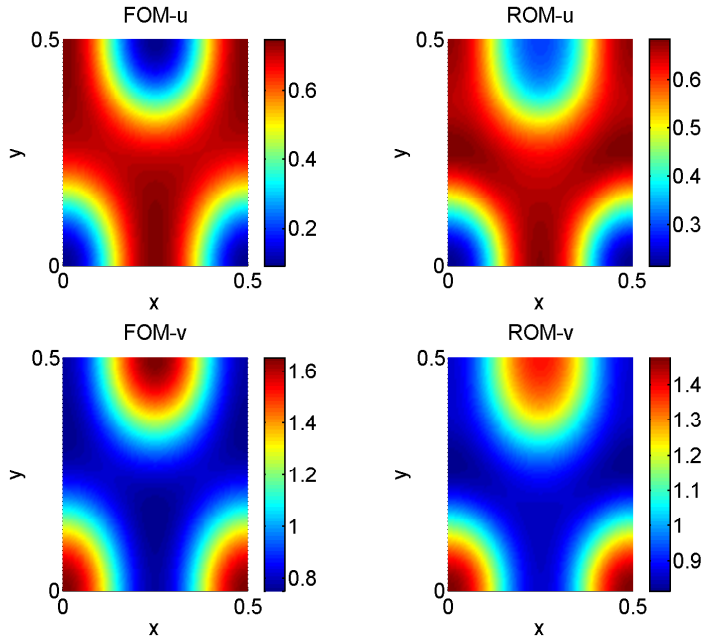


Figure 3: Schnakenberg model: FOM and ROM patterns at the final time for $\theta = 0.65$

In case of computational efficiency, the ROM approximations are obtained by a speed-up factor 20 over the FOM, Table 1. According to the energy criteria (28), it requires global factor matrices of column size (number of modes) only 8 – 11. The computed time averaged relative errors defined in (26) scales with 10^{-2} . Detailed results for the number of modes and errors are presented in Table 2.

4.2 Brusselator equation

We consider the three-dimensional Brusselator model [6] in the cubic domain $\Omega = [0, 20]^3 \subset \mathbb{R}^3$. We solve the problem through the matrix equation (16) with the mesh sizes $h_x = h_y = h_z = 0.667$ for the number of grid points $n_1 = n_2 = n_3 = 31$. The final time is taken as $t_f = 10$ with the time step size $\Delta t = 0.01$, leading to the fourth dimension $n_4 = 1001$ of the snapshot tensors related to the time. For this problem, we fix the system parameters $d_u = 0.4$, $d_v = 2$, $d_{uv} = 0.02$, $\alpha = 6$, $\beta = 1$, and vary now the parameter $\theta := d_{vu}$ in the set of admissible values $P = [19, 23]$. As the finite training set of parameter θ , we take the values (including the boundary values) uniformly distributed on P with the increment 1, i.e., $P_T = \{19, 20, 21, 22, 23\}$ with the number of sample parameter values $n_p = 5$.

In Figure 4, we give the decay of the singular values $\sigma_i(\mathcal{X}_{(j)}^{\theta,u})$ and $\sigma_i(\mathcal{X}_{(j)}^{\theta,v})$ of the unfoldings $\mathcal{X}_{(j)}^{\theta,u}$ and $\mathcal{X}_{(j)}^{\theta,v}$ of the order-4 snapshot tensors $\mathcal{X}^{\theta,u}$ and $\mathcal{X}^{\theta,v}$, respectively, $j = 1, 2, 3, 4$, related to each sample parameter value $\theta \in P_T$. According to the criteria (27), the computed target ranks $r_j^{\theta,u}$ and $r_j^{\theta,v}$ required by the HOSVD in the first level of nested POD are presented in Figure 5. Similar to the previous example, it again shows that enough energetic part of the unfoldings are recovered.

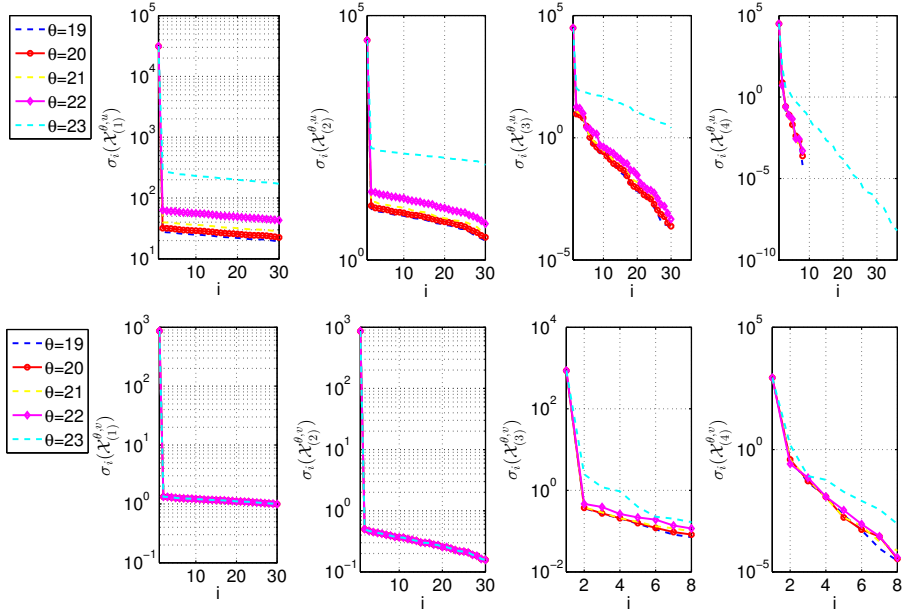


Figure 4: Brusselator model: Level I singular values of unfoldings $\mathcal{X}_{(j)}^{\theta,u}$ (top) and $\mathcal{X}_{(j)}^{\theta,v}$ (bottom)

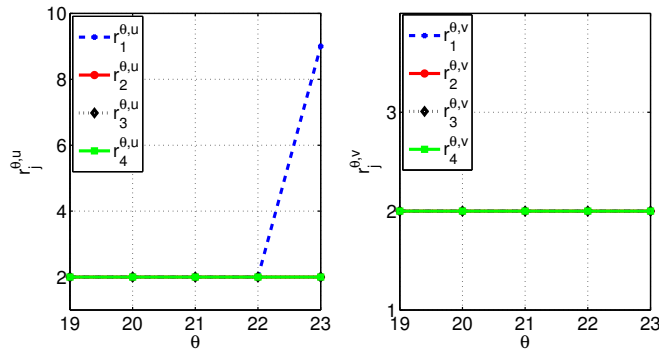


Figure 5: Brusselator model: Sample parameter values vs target ranks for unfoldings $\mathcal{X}_{(j)}^{\theta,u}$ (left) and $\mathcal{X}_{(j)}^{\theta,v}$ (right) at Level I

The FOM solutions $u(\mathbf{x}, t; \theta)$ and $v(\mathbf{x}, t; \theta)$ together with the nonintrusive ROM approximations $\hat{u}(\mathbf{x}, t; \theta)$ and $\hat{v}(\mathbf{x}, t; \theta)$ at the final time $t_f = 10$ for the parameter value $\theta = 21.5 \notin P_T$ are given in Figure 6, where it can be seen that enough similar patterns are obtained.

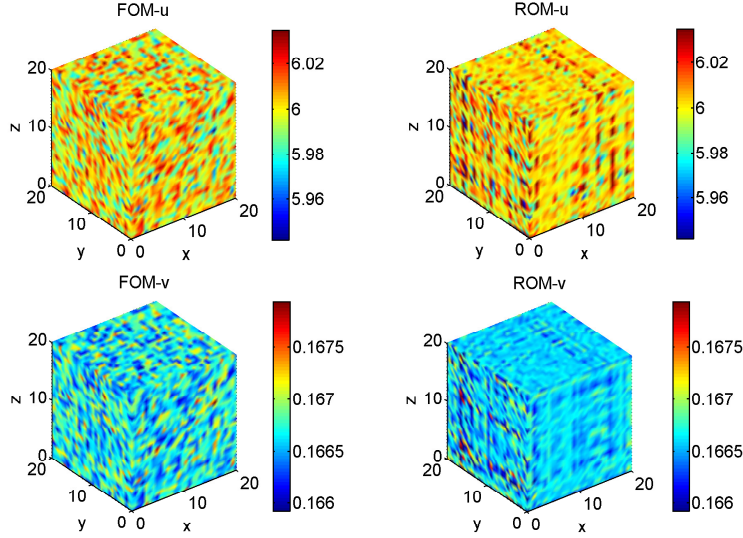


Figure 6: Brusselator model: FOM and ROM profiles at the final time for $\theta = 21.5$

In case of computational efficiency, Table 1 shows that the ROM approximations are obtained by a much greater speed-up factor, 130, over the FOM compared with the speed-up factor obtained for the two-dimensional Schnakenberg model. According to the energy criteria (28), it requires global factor matrices of column size only 6 – 14. The computed time averaged relative errors defined in (26) scales with 10^{-3} . The detailed results for the Brusselator model can also be found in Table 2.

Table 1: Wall clock time (in seconds) and speed-up factors

			Wall Clock Time	Speed-up
Schnakenberg	Offline	FOMs	53.70	
		Level I HOSVD Modes	223.29	
		Level II POD Modes	0.03	
		RBF Coefficients	0.93	
	Online ($\theta = 0.65$)	FOM	13.85	
	ROM	0.70	19.7	
Brusselator	Offline	FOMs	586.77	
		Level I HOSVD Modes	74.70	
		Level II POD Modes	0.01	
		RBF Coefficients	0.93	
	Online ($\theta = 21.5$)	FOM	123.30	
	ROM	0.95	129.8	

Table 2: Time averaged relative errors and memory savings of compression

		#Modes (u, v)	$\ u - \hat{u}\ _{\text{rel}}$	$\ v - \hat{v}\ _{\text{rel}}$	Saved Memory in %
Schnakenberg ($\theta = 0.65$)	x -direction (\hat{r}_1)	11, 10			
	y -direction (\hat{r}_2)	10, 9	9.54e-02	8.19e-02	%99
	t -direction (\hat{r}_3)	9, 8			
Brusselator ($\theta = 21.5$)	x -direction (\hat{r}_1)	14, 7			
	y -direction (\hat{r}_2)	7, 7			
	z -direction (\hat{r}_3)	7, 7	7.07e-03	6.63e-03	%99
	t -direction (\hat{r}_4)	6, 6			

We finally report the computational efficiency of the ST-HOSVD over T-HOSVD. To do this, we consider the order-3 and order-4 snapshot tensors $\mathcal{X}^{\theta_1, u}$ related to the component u of both the two-dimensional Schnakenberg and three-dimensional Brusselator models with the same problem data considered above. We apply T-HOSVD and ST-HOSVD to the snapshot tensors $\mathcal{X}^{\theta_1, u}$ with different values of target ranks $r := r_1^{\theta_1, u} = \dots = r_{d+1}^{\theta_1, u}$, and with the processing order $\mathbf{p} = [1, \dots, d + 1]$ for the ST-HOSVD. In Figure 7, we give the wall-clock times elapsed to make the SVD computations for each unfolding $\mathcal{X}_{(j)}^{\theta_1, u}$ of the order- $(d + 1)$ snapshot tensors $\mathcal{X}^{\theta_1, u}$, $d = 2, 3$. It is clear that ST-HOSVD provides, in total, better computational efficiency compared to the T-HOSVD. Moreover, as the algorithm progresses, the time needed for the SVD computation decreases in ST-HOSVD, while it remains almost the same in the case of T-HOSVD. This is because in the ST-HOSVD the dimension reduction is done sequentially, where the unfoldings are always obtained from the same full-rank snapshot tensor in T-HOSVD.

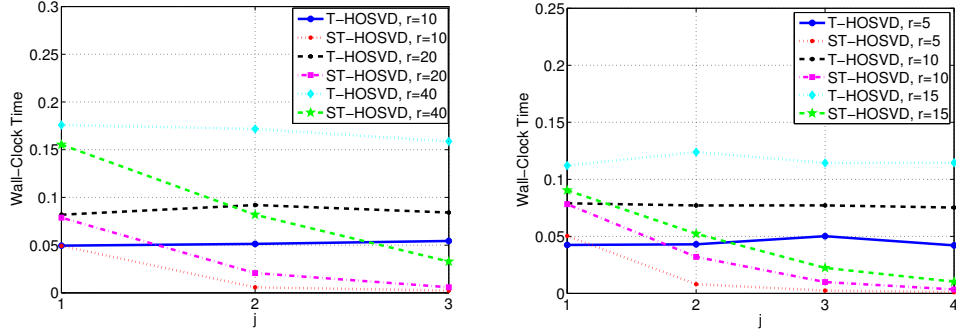


Figure 7: Wall-clock times of SVD computations elapsed for each unfolding $\mathcal{X}_{(j)}^{\theta_1, u}$ of the snapshot tensor $\mathcal{X}^{\theta_1, u}$ of Schnakenberg model (left) and Brusselator model (right), inside T-HOSVD and ST-HOSVD with different target rank r

5 Conclusions

In this paper, we have developed nonintrusive ROMs exploiting the matrix/tensor based discretization of cross-diffusion systems in form of semilinear PDEs. The two-level approach for the construction of reduced bases through tensor decompositions with HOSVD instead of the classical SVD yields the reduced modes and reduced coefficients directly without necessitating further computation in the case of the vector based discretization. Numerical experiments with two-dimensional and three-dimensional cross-diffusion systems demonstrate the computational efficiency of the ROMs and the accuracy of the spatiotemporal patterns for new parameter values.

References

- [1] G. Amitava, P. Jain, S. Kar, Alteration in cross diffusivities governs the nature and dynamics of spatiotemporal pattern formation, *ChemPhysChem* 21 (14) (2020) 1608–1616. doi:10.1002/cphc.202000142.
- [2] N. An, X. Yu, C. Huang, M. Duan, Local discontinuous Galerkin methods coupled with implicit integration factor methods for solving reaction-cross-diffusion systems., *Discrete Dynamics in Nature & Society* (2016) 1 – 18.
- [3] M. Dehghan, M. Abbaszadeh, Variational multiscale element free galerkin (VMEFG) and local discontinuous galerkin (LDG) methods for solving two-dimensional brusselator reaction–diffusion system with and without cross-diffusion, *Computer Methods in Applied Mechanics and Engineering* 300 (2016) 770 – 797. doi:10.1016/j.cma.2015.11.033.
- [4] G. Gambino, M. C. Lombardo, S. Lupo, M. Sammartino, Super-critical and sub-critical bifurcations in a reaction-diffusion Schnakenberg model with linear cross-diffusion, *Ricerche di Matematica* 65 (2) (2016). doi:10.1007/s11587-016-0267-y.
- [5] A. Madzvamuse, H. S. Ndakwo, R. Barreira, Cross-diffusion–driven instability for reaction–diffusion systems: analysis and simulations, *Journal of Mathematical Biology* 70 (4) (2015) 709–743. doi:10.1007/s00285-014-0779-6.

- [6] Z. Lin, R. Ruiz-Baier, C. Tian, Finite volume element approximation of an inhomogeneous Brusselator model with cross-diffusion, *Journal of Computational Physics* 256 (2014) 806–823. doi:10.1016/j.jcp.2013.09.009.
- [7] J. Zhang, G. Yan, Lattice Boltzmann simulation of pattern formation under cross-diffusion, *Computers & Mathematics with Applications* 69 (3) (2015) 157–169. doi:10.1016/j.camwa.2014.11.016.
- [8] G. Q. Sun, Z. Jin, L. Li, M. Haque, B. L. Li, Spatial patterns of a predator-prey model with cross diffusion, *Nonlinear Dynamics* 69 (4) (2012) 1631–1638. doi:10.1007/s11071-012-0374-6.
- [9] E. Tulumello, M. C. Lombardo, M. Sammartino, Cross-diffusion driven instability in a predator-prey system with cross-diffusion, *Acta Applicandae Mathematicae* 132 (1) (2014) 621–633.
- [10] V. Simoncini, Computational methods for linear matrix equations, *SIAM Review* 58 (3) (2016) 377–441. doi:10.1137/130912839.
- [11] D. Palitta, V. Simoncini, Matrix-equation-based strategies for convection–diffusion equations, *BIT Numerical Mathematics* 56 (2) (2016) 751–776. doi:10.1007/s10543-015-0575-8.
- [12] M. C. D’Autilia, I. Sgura, V. Simoncini, Matrix-oriented discretization methods for reaction-diffusion PDEs: Comparisons and applications, *Computers & Mathematics with Applications* 79 (7) (2020) 2067–2085. doi:10.1016/j.camwa.2019.10.020.
- [13] V. Simoncini, Numerical solution of a class of third order tensor linear equations, *Bollettino dell’Unione Matematica Italiana* 13 (3) (2020) 429–439. doi:10.1007/s40574-020-00247-4.
- [14] G. Berkooz, P. Holmes, J. L. Lumley, The proper orthogonal decomposition in the analysis of turbulent flows, *Annual Review of Fluid Mechanics* 25 (1) (1993) 539–575. doi:10.1146/annurev.fl.25.010193.002543.
- [15] L. Sirovich, Turbulence and the dynamics of coherent structures. III. Dynamics and scaling, *Quarterly of Applied Mathematics* 45 (3) (1987) 583–590. doi:10.1090/qam/910464.
- [16] S. Chaturantabut, D. C. Sorensen, Nonlinear model reduction via discrete empirical interpolation, *SIAM Journal on Scientific Computing* 32 (5) (2010) 2737–2764.
- [17] B. Karasözen, M. Uzunca, T. Küçükseyhan, Model order reduction for pattern formation in FitzHugh–Nagumo equations, in: B. Karasözen, M. Manguoğlu, M. Tezer-Sezgin, S. Göktepe, Ö. Uğur (Eds.), *Numerical Mathematics and Advanced Applications ENUMATH 2015*, Springer International Publishing, Cham, 2016, pp. 369–377.
- [18] B. Karasözen, G. Mülayim, M. Uzunca, S. Yıldız, Reduced order modelling of nonlinear cross-diffusion systems, *Applied Mathematics and Computation* 401 (2021) 126058. doi:10.1016/j.amc.2021.126058.
- [19] C. Audouze, F. De Vuyst, P. B. Nair, Nonintrusive reduced-order modeling of parametrized time-dependent partial differential equations, *Numerical Methods for Partial Differential Equations* 29 (5) (2013) 1587–1628. doi:10.1002/num.21768.
- [20] W. Chen, J. S. Hesthaven, B. Junqiang, Y. Qiu, Y. Tihao, Z. Yang, Greedy non-intrusive reduced order model for fluid dynamics, *AIAA Journal* 56 (2018) 12. doi:10.2514/1.J056161.
- [21] D. Xiao, F. Fang, C. Pain, G. Hu, Non-intrusive reduced-order modelling of the navier–stokes equations based on RBF interpolation, *International Journal for Numerical Methods in Fluids* 79 (11) (2015) 580–595. doi:10.1002/flid.4066.
- [22] L. De Lathauwer, B. De Moor, J. Vandewalle, A multilinear singular value decomposition, *SIAM Journal on Matrix Analysis and Applications* 21 (4) (2000) 1253–1278. doi:10.1137/S0895479896305696.
- [23] N. Vannieuwenhoven, R. Vandebril, K. Meerbergen, A new truncation strategy for the higher-order singular value decomposition, *SIAM Journal on Scientific Computing* 34 (2) (2012) A1027–A1052. doi:10.1137/110836067.
- [24] M. K. Moayyedi, M. N. Beygi, A high fidelity cost efficient tensorial method based on combined POD-HOSVD reduced order model of flow field, *European Journal of Computational Mechanics* 27 (4) (2018) 342–366. doi:10.1080/17797179.2018.1550963.
- [25] Y. Choi, K. Carlberg, Space–time least-squares Petrov–Galerkin projection for nonlinear model reduction, *SIAM Journal on Scientific Computing* 41 (1) (2019) A26–A58. doi:10.1137/17M1120531.
- [26] G. Kirsten, V. Simoncini, A matrix-oriented POD-DEIM algorithm applied to nonlinear differential matrix equations, *ArXiv preprint: 2006.13289* (2021).
- [27] L. R. Tucker, Some mathematical notes on three-mode factor analysis, *Psychometrika* 31 (3) (1966) 279–311. doi:10.1007/BF02289464.

-
- [28] R. Minster, A. K. Saibaba, M. E. Kilmer, Randomized algorithms for low-rank tensor decompositions in the Tucker format, *SIAM Journal on Mathematics of Data Science* 2 (1) (2020) 189–215. doi:10.1137/19M1261043.
- [29] L. S. Lorente, J. M. Vega, A. Velazquez, Compression of aerodynamic databases using high-order singular value decomposition, *Aerospace Science and Technology* 14 (3) (2010) 168–177. doi:10.1016/j.ast.2009.12.003.