MASTER

Model reduction of multidimensional dynamical systems by tensor decompositions

Achterberg, J.

Award date:
2008

Link to publication
Model reduction of multidimensional dynamical systems by tensor decompositions

by

Jaron Achterberg

Master of Science thesis

Project period: 2008
Report Number: 08A/05

Commissioned by:
Prof.dr.ir. A.C.P.M. Backx

Supervisors:
Dr. S. Weiland
Dr.ir. J. Ludlage (IPCOS)
Model reduction of multidimensional dynamical systems by tensor decompositions

Jaron Achterberg

Abstract—This paper investigates different methods for multi dimensional POD reduction. The POD basis functions will be obtained from a tensor decomposition method. Multiple methods are available and will be compared for their use in model reduction. These methods will be applied to a dynamical system described by Partial Differential Equations, PDEs. The methods are well defined on functions on Cartesian domains, several methods to apply them on non Cartesian domains are treated as well.

I. INTRODUCTION

An important part of scientific research is dedicated to the creation of mathematical models of physical processes. A lot of effort is made to describe dynamical processes as accurate as possible. Control designers use these models to find a suitable controller. However, often the physical processes that need to be controlled are very complex and, as a consequence, result in large scale models. Many modern controller synthesis methods will create a controller with equal order as the order of the plant model. For complex processes this results in controllers that will be hard to implement or whose operation is restricted by limited computational power.

To achieve a technically feasible controller for complex systems one has either to reduce the model of the plant or the order of the controller. Various reduction methods have already been developed, that are able to handle different classes of models. Physical processes that are described by Partial Differential Equations, (PDEs) are known to be computationally intensive for simulation. A Method for reduction of PDE models was developed in [1]. This method is primarily known for this application as Proper Orthogonal Decomposition, POD, and in other fields as, Karhunen Loève decomposition or Principal Component Analysis. For systems that evolve both over time as well as space, this method aims to decompose temporal-spatial signals in such a way that truncated spectral decompositions define optimal signal approximations in a well defined sense. Specifically, this method finds a spatial and temporal basis that can approximately describe the dynamics of the process.

This paper will look for an appropriate extension of method that uses the spatial structure to achieve a better reduction. The separation of the spatial structure follows easily for functions on a Cartesian domain. Yet to apply this method to functions on Non-Cartesian domains is a non trivial extension of the POD method that explicitly uses Cartesian structures in the spatial domain.

II. PROBLEM FORMULATION

A. Notation and terminology

Both superscript and subscript symbols have to be interpreted as indices. Functions are denoted by $u$, vectors by $a$ matrices by capital symbols $A$, the set of reals by $\mathbb{R}$ and more general sets by $\mathcal{A}$. Sets of functions will be written with calligraphic symbols $\mathcal{A}$. All inner products by $\langle \cdot, \cdot \rangle$.

Time derivatives, $\frac{df}{dt}$ are written by $\dot{f}$, partial derivatives $\frac{\partial f}{\partial x}$ and $\frac{\partial^2 f}{\partial x^2}$ by $f_x$ and $f_{xx}$ respectively.

B. Problem formulation

Many physical systems admit useful representations by Partial Differential Equations, PDEs. In this paper the problem that is addressed is to find given a PDE a reduced model of the system such that symmetries in the Cartesian structure of the domain is invariant in the reduction. In problems that have spatial domain of dimension larger than one, one can try to separate the spatial basis. The reduced model should approximate the original as close as possible. First spectral decompositions that admit separated basis functions have to be introduced, and reduction in this expansion is treated. To obtain the separated basis functions from data, the data will be described as a tensor so one can apply tensor decompositions. Several tensor decompositions are available and will be compared for their use in model reduction. Optimality of the reduction is investigated.

The focus will be on linear PDEs on Cartesian domains, so the complexity of the benchmark problems is bounded. The following example of a PDE is used throughout the paper to clarify the different methods that will be investigated.

Example 1 (Heat equation). The heat $u$ on a two dimensional slab material with dimensions $X \times Y = [0,L_x] \times [0,L_y]$ changes in time $u(t,x,y) : X \times Y \times T \mapsto \mathbb{R}$, according to the PDE,

$$\rho C \partial_t u = \kappa_x u_{xx} + \kappa_y u_{yy} + \nu_x u_x + \nu_y u_y.$$  

Here $\rho$ is the heat resistance and $C$ the heat capacity of the material, the parameters $\kappa$ and $\nu$ denote the conductivity and the viscosity of the material in the various directions.

The POD methods that are investigated all look for a finite decomposition of $u$ in components that are simple but still represent the original solution. These components are sorted in how well they resemble the original solution of the PDE in energy sense. Thus it becomes easy to determine which and
III. SPECTRAL DECOMPOSITIONS

To be able to reduce a large model, the model and its solution will be decomposed, such that it will be easy to find the dominant spectral components of the model and its solution. The reduction will follow easily from the truncation of the spectral decomposition. This method is well known for functions of one spatial dimension. Under certain conditions it is possible to extend this decomposition to a separation of functions on multidimensional domains in to multiple functions in the separated coordinates of the domain.

Like Fourier series where a function or signal is decomposed as the sum of trigonometric functions, it is sometimes possible to decompose a general function on any full basis according to

\[ f(x) = \sum_{i} a_i \phi^i(x). \]

Here \( \phi^i \) are called "basis functions", the (real) numbers \( a_i \) are coefficients. More precise,

Theorem 1. If \((\mathcal{X}, \langle \cdot, \cdot \rangle)\) is a separable Hilbert space with orthonormal basis \( \{ \phi_i \}_{i \in \mathbb{I}} \) any element \( f \in \mathcal{X} \) can be written as,

\[ f = \sum_{i \in \mathbb{I}} a_i \phi^i, \]

where \( a_i = \langle f, \phi^i \rangle \) are called Fourier or modal coefficients.

For the proof see, [2].

Consider the \( L_2(\mathbb{X}) \) inner product space on real functions \( \mathbb{X} \to \mathbb{R} \). For \( t \in \mathbb{T} \) fixed, assume \( f(t, \cdot), \phi^i \in L_2(\mathbb{X}) \), so that applying Theorem 1 with \( \mathcal{X} = L_2(\mathbb{X}) \) gives,

\[ a_i(t) = \langle f(t, \cdot), \phi^i \rangle = \int_{\mathbb{X}} f(t, x) \phi^i(x) dx. \]

This gives decomposition,

\[ f(t, x) = \sum_i a_i(t) \phi^i(x). \]

Functions of multitudes variables can also be decomposed. Introduce the Hilbert spaces, \( L_2(\mathbb{X}), \mathbb{Ω} \subseteq \mathbb{R}^d \) of functions \( f : \mathbb{Ω} \to \mathbb{R} \) with the inner product \( \langle f, g \rangle = \int_{\mathbb{Ω}} f(\omega)g(\omega)d\omega \). With the domain \( \mathbb{Ω} = \mathbb{X}, \mathbb{Y}, \mathbb{Z} = \mathbb{X} \times \mathbb{Y} \), we obtain,

\[ \mathcal{X} = L_2(\mathbb{X}), \]

\[ \mathcal{Y} = L_2(\mathbb{Y}), \]

\[ \mathcal{T} = L_2(\mathbb{T}), \]

\[ \mathcal{F} = L_2(\mathbb{X} \times \mathbb{Y}). \]

Let \( \{ \phi^i \} \) be an orthonormal basis of \( \mathcal{F} \), \( \{ \xi^j \} \) an orthonormal basis of \( \mathcal{X} \), \( \{ \eta^k \} \) an orthonormal basis of \( \mathcal{Y} \) and \( \{ \tau^k \} \) an orthonormal basis of \( \mathcal{T} \). Applying Theorem 1 with these Hilbert spaces leads to different decompositions:

\[ f(t, x, y) = \sum_{i=1}^{\infty} a_i(t) \phi^i(x, y), \]

\[ f(t, x, y) = \sum_{i=1}^{\infty} \sum_{j=1}^{\infty} a_{ij}(t) \xi^i(x) \eta^j(y), \]

\[ f(t, x, y) = \sum_{i=1}^{\infty} \sum_{j=1}^{\infty} \sum_{k=1}^{\infty} a_{ijk}(t) \tau^k(t) \xi^i(x) \eta^j(y). \]

The first decomposition, (4), follows directly from the Hilbert \( \mathcal{F} \) space with an inner product like (1), but now \((x, y)\) is substituted in \( x \).

To obtain the second decomposition (5), first Lemma 1 is introduced. For the proof see Appendix A. By use of Theorem 1, \( f(t, x, y) \) can be decomposed as,

\[ f = \sum_{i} \langle f, \xi^i \rangle \xi^i(x) = \sum_{i} \sum_{j} \langle \langle f, \xi^i \rangle \eta^j \rangle \eta^j(y) \xi^i(x). \]

By use of Lemma 1 this is the same as,

\[ = \sum_{i} \sum_{j} \langle f, \eta^j \rangle \xi^i(x) \eta^j(y) = \sum_{i} \sum_{j} a_{ij}(t) \xi^i(x) \eta^j(y). \]

thus \( a_{ij}(t) \) are uniquely defined by,

\[ a_{ij}(t) = \langle f, \eta^j \rangle \xi^i(x) = \langle \langle f, \xi^i \rangle \eta^j \rangle. \]

The third decomposition (6) is obtained in the same way, but now \( f \) is also projected on the basis \( \{ \tau^k \} \) of \( \mathcal{T} \).

A. Signal reduction

Reduction of the signal space is now achieved by truncation of the spectral expansion. Thus the representation of \( u \), can be reduced to an \( r \)-dimensional approximation by,

\[ u_r = \sum_{i=1}^{r} a_i \phi^i. \]

where, \( a_i \) and \( \phi^i \) are obtained from the spectral expansion of the original signal, \( u \). In the multi-variable case it is possible to have a different truncation order for each expansion argument. So, by use of equations (4)(5)(6), \( u(t, x, y) \) can be reduced in the following ways,

\[ u_r = \sum_{i=1}^{r} a_i(t) \phi^i(x, y). \]

\[ \phi^i \in \mathcal{F}_r \subset \mathcal{F}, \]
\[ u_{r_x, r_y} = \sum_{i=1}^{r_x} \sum_{j=1}^{r_y} a_{ij}(t) \xi^i(x) \eta^j(y) \]
\[ u_{r_t, r_x, r_y} = \sum_{i=1}^{r_t} \sum_{j=1}^{r_x} \sum_{k=1}^{r_y} a_{ijk} \tau^i(t) \xi^j(x) \eta^k(y) \]
\[ \xi^i \in \mathcal{X}_r \subset \mathcal{X}, \]
\[ \eta^j \in \mathcal{Y}_r \subset \mathcal{Y}, \]
\[ \tau^i \in \mathcal{T}_r \subset \mathcal{T}. \]

Here \( \mathcal{F}, \mathcal{X}, \mathcal{Y}, \mathcal{T} \) are the same spaces as given in (3).

### B. System reduction

For reduction of the system the residual, \( R(u) \) of the PDE has to be considered. In the case of Example 1 the residual is,
\[ R(u) := \rho C u_t - \kappa_x u_{xx} - \kappa_y u_{yy} - \nu_x u_x - \nu_y u_y. \]
Let \( \mathcal{H} \) be a Hilbert space such that \( R : \text{dom}(R) \rightarrow \mathcal{H} \) is a well defined map. Call \( u \) a weak solution of the PDE if,
\[ \langle R(u), \phi \rangle_{\mathcal{H}} = 0, \quad \text{for all } \phi \in \mathcal{H}. \]

Now reduction of the system means that the weak solutions are relaxed to the requirement that
\[ \langle R(u), \phi \rangle_{\mathcal{H}_r} = 0, \quad \text{for all } \phi \in \mathcal{H}_r \subset \mathcal{H}, \quad (7) \]
where \( \mathcal{H}_r \) is a \( r \)-dimensional subspace of \( \mathcal{H} \). Depending on the choices of \( \mathcal{H} \) and \( \mathcal{H}_r \) this gives different reduced order models,

1. \( r = r_{xy} \)
   - \( \mathcal{H} = \mathcal{F} \)
   - \( \mathcal{H}_r = \text{span}\{\phi_1, \ldots, \phi_r\} \)
   - \( 0 = \langle R(u), \phi \rangle_{\mathcal{H}_r} \quad \text{for all } \phi \in \mathcal{H}_r \subset \mathcal{H}. \)

2. \( r = (r_x, r_y) \)
   - \( \mathcal{H} = \mathcal{X} \otimes \mathcal{Y} \)
   - \( \mathcal{H}_r = \text{span}\{\xi_1, \ldots, \xi_{r_x}\} \times \text{span}\{\eta_1, \ldots, \eta_{r_y}\} \)
   - \( 0 = \langle \langle R(u), \xi \rangle_{\mathcal{X}}, \eta \rangle_{\mathcal{Y}} \quad \text{for all } \xi \in \mathcal{X}_r \subset \mathcal{X}, \eta \in \mathcal{Y}_r \subset \mathcal{Y}. \)

3. \( r = (r_t, r_x, r_y) \)
   - \( \mathcal{H} = \mathcal{T} \otimes \mathcal{X} \otimes \mathcal{Y} \)
   - \( \mathcal{H}_r = \text{span}\{\tau_1, \ldots, \tau_{r_t}\} \times \text{span}\{\xi_1, \ldots, \xi_{r_x}\} \times \text{span}\{\eta_1, \ldots, \eta_{r_y}\} \)
   - \( 0 = \langle \langle \langle R(u), \xi \rangle_{\mathcal{X}}, \eta \rangle_{\mathcal{Y}}, \tau \rangle_{\mathcal{T}} \quad \text{for all } \tau \in \mathcal{T}_r \subset \mathcal{T}, \xi \in \mathcal{X}_r \subset \mathcal{X}, \eta \in \mathcal{Y}_r \subset \mathcal{Y}. \)

### C. Galerkin Projection

For controller design it is useful to convert these PDEs into a set of Ordinary Differential Equations, ODEs. A method to achieve this is by Galerkin projection [3]. If \( u \) has a spectral decomposition \( u = \sum_k a_k \phi^k \) with \( \phi^k \) orthonormal basis functions in a Hilbert space \( \mathcal{H} \) and \( \mathcal{H}_r = \text{span}\{\phi^1, \ldots, \phi^r\} \) then the reduced order model is equivalent to an ODE in the coefficient vector \( \alpha = [a_1, \ldots, a_r]^T \) of particular simple structure. For Example 1 this can be obtained for the expansion of (3).
\[ 0 = \langle R(u), \phi^k(x, y) \rangle, \]
is equivalent to,
\[ 0 = \langle \rho C u_t - \kappa_x u_{xx} - \kappa_y u_{yy} - \nu_x u_x - \nu_y u_y, \phi^k(x, y) \rangle. \]
Reduction of the solution, \( u_r(t, x, y) = \sum_{i=1}^{r} a_i(t) \phi^i(x, y), \)
\[ \left\{ \rho C \sum_{i=1}^{r} \dot{a}_i(t) \phi^i(x, y) - \kappa_x \sum_{i=1}^{r} a_i(t) \phi^i_{xx}(x, y) \right. \]
\[ - \kappa_y \sum_{i=1}^{r} a_i(t) \phi^i_{yy}(x, y) - \nu_x \sum_{i=1}^{r} a_i(t) \phi^i_x(x, y) \]
\[ - \nu_y \sum_{i=1}^{r} a_i(t) \phi^i_y(x, y), \phi^k(x, y) \right\} = 0. \]
This leads to,
\[ \rho C \dot{a}_k(t) - \sum_{i=1}^{r} \left( \kappa_x \phi^i_{xx} - \kappa_y \phi^i_{yy} - \nu_x \phi^i_x - \nu_y \phi^i_y, \phi^k \right) a_i(t) = 0. \]
With \( \mathbf{a}(t) = [a_1(t), \ldots, a_r(t)]^T \), this becomes an explicit ODE in the vector \( \mathbf{a}, \)
\[ \rho C \dot{\mathbf{a}} = A \mathbf{a}, \]
where,
\[ A_{ij} = \left( \kappa_x \phi^i_{xx} - \kappa_y \phi^i_{yy} - \nu_x \phi^i_x - \nu_y \phi^i_y, \phi^k \right) \quad i, j = 1, \ldots, r. \]
The tensor decomposition methods use an expansion as given in (5), so in the same way a set of ODEs will be obtained from the PDE of Example 1 but now with this expansion,
\[ \langle \rho C \dot{u}, \xi^k, \eta^j \rangle = \langle \kappa_x u_{xx} + \kappa_y u_{yy} + \nu_x u_x + \nu_y u_y, \xi^k, \eta^j \rangle. \]
Reduction of the solution, \( \dot{u}(t, x, y) = \sum_{i=1}^{r_x} \sum_{j=1}^{r_y} a_{ij}(t) \xi^i(x) \eta^j(y) \) leads to
\[ \rho C \sum_{j=1}^{r_y} \sum_{i=1}^{r_x} a_{ij}(t) \xi^i(x) \eta^j(y) \]
\[ \left\{ \rho C \sum_{j=1}^{r_y} \dot{a}_{kj} \eta^j \right\} = \left\{ \sum_{i=1}^{r_x} \sum_{j=1}^{r_y} a_{ij}(t) \kappa_x \xi^i_{xx} + \nu_x \xi^i_x + \kappa_y \xi^i_{yy} + \nu_y \xi^i_y, \eta^j \right\} + \left\{ \sum_{j=1}^{r_y} \dot{a}_{kj} \eta^j \right\}, \]
\[ \rho C \dot{a}_{kl} = \sum_{i=1}^{r_x} \left( \kappa_x \xi^i_{xx} + \nu_x \xi^i_x \right) a_{il} + \sum_{j=1}^{r_y} \left( \kappa_y \eta^j_{yy} + \nu_y \eta^j_y \right) a_{lj}. \]
With \( \langle A \rangle_{kl}(t) = a_{kl} \) this can be expressed as,
\[ \rho C \dot{\mathbf{a}} = \left( \kappa_x \mathbf{E}_{xx} + \nu_x \mathbf{E}_x \right) \mathbf{a} + \mathbf{A} \left( \kappa_y H_{yy} + \nu_y H_y \right). \]
This is again an explicit ODE, now in the matrix $A$.

For the decomposition of $\mathcal{G}$ the set of equations results in a pure Algebraic Equation of the tensor $A \in \mathbb{R}^{r_1 \times r_2 \times \cdots \times r_N}$. Solving this Algebraic Equation will give a result that is static which is not so useful for it can not be used as a dynamic system, therefore the details are omitted.

IV. TENSOR DECOMPOSITIONS ON CARTESIAN DOMAINS

In the previous section the spectral decompositions \[4, 5, 6\], were introduced. It was also shown how PDEs can be transformed to ODEs by performing Galerkin Projections. However, the basis functions used in the expansion and the projections have not been specified. In Fourier series expansions sine and cosine functions are used as basis functions. But in general any set of orthonormal functions could be used. To achieve an optimal approximation by truncated spectral decompositions, the POD method proposes using basis functions derived from data. Measured or simulated data of a position and time dependent function is stored as snapshots in a matrix, such that the time samples, which are vectors containing spatial information, are represented by the columns. Applying the SVD to the matrix, gives besides the singular values two unitary matrices, a matrix whose columns represent spatial basis vectors and another whose columns represent the temporal basis. The basis vectors are ordered and can be used as basis functions for spectral decompositions and projections.

In problems that have a larger than one dimensional spatial domain or range, one could also try to separate the spatial basis. In general these kind of functions can be represented by a multidimensional linear mapping from $M$ to $N$ function spaces, $\mathcal{X}_1 \times \cdots \times \mathcal{X}_M \rightarrow \mathcal{Y}_1 \times \cdots \times \mathcal{Y}_N$.

The mathematical object to describe mappings between multiple vector spaces is the tensor. A multi dimensional linear mapping toward $\mathbb{R}$, that maps the spaces of the domain and the dual spaces of the range toward $\mathbb{R}$, $\mathcal{X}_1 \times \cdots \times \mathcal{X}_M \times \mathcal{Y}_1^* \times \cdots \times \mathcal{Y}_N^* \rightarrow \mathbb{R}$. The challenge is now to find a suitable decomposition of this tensor such that reduction of the dimension of the basis is optimal. Some properties and terminology of a tensor that will be used in the treatment of the different tensor decompositions will be given. Several of these terms are well known in literature \[4\] others like tensor rank are specifically defined for the use in tensor decompositions.

Definition 1 (Tensor). A tensor is a multi linear functional, $T : \mathcal{Y}_1 \times \cdots \times \mathcal{Y}_N \rightarrow \mathbb{R}$.

The number of vector spaces the tensor operates on is called the order of the tensor. The space of order-$N$ tensors is a vector space, $\mathcal{T}_N$. Addition and scalar multiplication is inherited from all the vector spaces the tensor operates on.

Example 2 (The matrix as an order 2 tensor). A matrix, $A \in \mathbb{R}^{L_x \times L_y}$ defines a tensor $A : \mathbb{R}^{L_x} \times \mathbb{R}^{L_y} \rightarrow \mathbb{R}$, $A(x, y) := \langle Ay, x \rangle = \langle A^T x, y \rangle$.

The tensor can be viewed as a generalisation of the outer product for vectors, $A = xx^T$.

Definition 2 (Tensor product). For elements $u_n \in \mathcal{V}_n$, $n = 1, \ldots, N$ the tensor product $u_1 \otimes \cdots \otimes u_N$ is the tensor:

$$(u_1 \otimes \cdots \otimes u_N)(v_1, \ldots, v_N) = \prod_{i=1}^{N} \langle u_i, v_i \rangle.$$

The tensor $U = u_1 \otimes \cdots \otimes u_N$ is called a rank-one tensor.

The tensor elements of $T \in \mathcal{T}_N$ are specified by $t_{i_1,\ldots,i_N}$ where $i_n$ ranges through $1, \ldots, \dim \mathcal{V}_n$. The elements $t_{i_1,\ldots,i_N}$ represent $T$ with respect to a special basis, $\{e_1^N, \ldots, e_N^N\}$, such that $t_{i_1,\ldots,i_N} = \langle T(e_1^N), \ldots, e_N^N \rangle$. The tensor can be decomposed on the natural unit basis $\{e_1^N \otimes \cdots \otimes e_N^N\}$, according to

$$T = \sum_{i_1}^{L_1} \cdots \sum_{i_N}^{L_N} t_{i_1,\ldots,i_N} (e_1^N \otimes \cdots \otimes e_N^N).$$

Definition 3 ($n$-mode kernel of a tensor $T$),

$$\ker_n T := \{v_n \in \mathcal{V}_n | T(v_1, \ldots, v_N) = 0, \forall v_k \in \mathcal{V}_k, k \neq n\}.$$  

In matrix theory the rank of a matrix equals the column and row rank of a matrix, which happens to be the same number. For tensors the notion of row or column rank can be seen as a modal rank.

Definition 4 (Modal rank). The tensor $T$ as $\mathcal{V}_1 \times \cdots \times \mathcal{V}_N \rightarrow \mathbb{R}$ has the $n$-mode rank of $T$ defined as $\dim \mathcal{V}_n - \dim (\ker_n(T))$. Notice the resemblance with the Rank-Nullity theorem for matrices.

The modal rank of a tensor is not the same in each mode, so the notion of overall tensor rank is difficult to express. Therefore a tensor of rank one, which is well defined, is used.

Definition 5 (Tensor Rank). The minimum number $R$ of rank one tensors that can reconstruct the original tensor like, $T = \sum_{i=1}^{R} U_i$, with $U_i$ of rank 1 is called the rank of the tensor $T$.

Although these concepts are quite intuitive extensions of the rank known from linear algebra, there are no general results, only some special cases\[5\], on the maximum rank a tensor of arbitrary order and dimensions can attain.

In the tensor definition the vector spaces on which $T$ operates were already assumed to have an inner product. Now the inner product on tensors can be defined.
Definition 6 (Tensor Inner product). Given two tensors \( S, T \in T_N \) define the inner product
\[
(T, S) := \sum_{i_1}^{L_1} \cdots \sum_{j_N}^{L_N} t_{i_1 \ldots i_N} s_{j_1 \ldots j_N} (e_i^{N})_{1}, \ldots, (e_i^{N})_{N},
\]
where \( \langle \cdot, \cdot \rangle \) is the inner product defined on \( V_i \). \( T_N \) equipped with \( \langle \cdot, \cdot \rangle \) becomes an inner product space.

There are multiple norms possible for a tensor:

Definition 7 (The Frobenius norm). The Frobenius norm can easily be defined by using the inner product,
\[
\|T\|_{F} = \sqrt{(T, T)}.
\]

Definition 8 (The operator norm).
\[
\|T\| = \sup_{\|v\|=1} |T(v_1, \ldots, v_N)|.
\]

In Appendix B the decomposition of functionals in a tensor setting is treated.

A. High Order SVD (HOSVD)

This decomposition was proposed by Lieven De Lathouwer [6], it uses the basis obtained from the normal matrix SVD on unfoldings of the tensor in all its directions. The \( i^{th} \) unfolding of a \( N \)-way array along its \( i^{th} \) mode is actually the reordering of its elements into a matrix with the dimensions, \( W_i \in \mathbb{R}^{L_i \times L_1 \ldots L_{i-1} L_{i+1} \ldots L_N} \). The \( k^{th} \) row in the \( i^{th} \) unfolding is the sequence of all elements in \( T(\ldots, e_i^k, \ldots, \ldots) \) in an arbitrary order. This is depicted for tensor of order three in Fig. 1.

In [7] it is stated that the rank one approximation made by the first singular vectors obtained from this decomposition is not the best possible. This is apart from that its definition is just on elements of the tensor instead of the operation the tensor represents another disadvantage. A great advantage though is that its algorithm uses the very well developed SVD for matrices.

Example 3. HOSVD on a simple rank \( 2 \times 2 \times 2 \) tensor
In Fig. 2(a) the tensor \( T(x, y, z) \) is depicted.
\[
T(x, y, z) = t_{ijk} x^i \otimes y^j \otimes z^k,
\]
with the coefficients \( t_{ijk} \).
\[
t_{111} = 1, \quad t_{121} = 0, \quad t_{112} = 1, \quad t_{122} = 0, \quad t_{211} = 1, \quad t_{221} = 0, \quad t_{212} = 0, \quad t_{222} = 0.
\]

This tensor has the unfoldings,
\[
W_x = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad W_y = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad W_z = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}.
\]

The SVD applied separately on these unfoldings gives basis elements defined by left singular vectors. \( W_i = U_i S_i V_i \), gives,
\[
U_x = \begin{pmatrix} -0.5257 & -0.8507 \\ -0.8507 & 0.5257 \end{pmatrix}, \quad U_y = \begin{pmatrix} 0 & 1 \\ 1 & -0 \end{pmatrix}, \quad U_z = \begin{pmatrix} -0.5257 & -0.8507 \\ -0.8507 & 0.5257 \end{pmatrix}.
\]

The core tensor \( S \), depicted in Figure 2(b) is obtained by,
\[
S_{i,j,k} = \langle (U_x e_i^x, (U_y e_j^y, (U_z e_k^z) \rangle.
\]

Example 4. HOSVD on the heat equation data. Data was obtained from the simulation of the heat equation of Example 1 simulated on a grid of \( 61 \times 81 \) spatial samples and over a time horizon of \([0, 3]\) with sample time 0.05s. It has an asymmetric initial condition, see Fig. 5 that is applied to the dynamics of Example 1 with the parameters,
\[
\kappa_x = 0.5, \quad \nu_x = 0.5, \quad \rho C = 5, \quad \kappa_y = 0.5, \quad \nu_y = 0.5.
\]

Application of the HOSVD algorithm resulted in a basis in \( x, y \) and \( t \) separately. In Fig. 4 the first five basis functions in both the \( x \) and \( y \) direction are shown. One can observe the displaced peaks of the different basis functions to cover the convective behaviour of the system.

B. Tensor SVD (TSVD)

This method has been proposed by Belzen, Weiland, de Graaf [8]. It looks for a set of orthonormal vectors in each domain, \( V_n, n = 1, \ldots, N \) that maximises the tensor such that the vectors are unit. The algorithm keeps searching for extra vectors such that they are orthogonal to the set that is already found, where orthonormality is determined by the inner products defined on the vector spaces the tensor operates on.

Definition 9 (Singular values).
\[
\sigma_1(T) = \|T\| = \langle v_1^1, \ldots, v_1^N, \rangle.
\]
The vectors $v^k_k = 1, \ldots, \dim V_n$ are now considered to be basis vectors of $V_n$. The N-th order tensor can be decomposed,

$$T = \sum_{k_1}^{L_1} \cdots \sum_{k_N}^{L_N} S_{k_1, \ldots, k_N} v_1^{k_1} \otimes \cdots \otimes v_N^{k_N},$$

where $S$, the core tensor, is obtained by projection of $T$ onto the basis. That is,

$$S_{k_1, \ldots, k_N} = T(v_1^{k_1}, \ldots, v_N^{k_N}).$$

Notice that the singular values $\sigma_k$ are on the diagonal $S_{k \ldots k}$. This optimisation is solved by solving for each $k$ consecutively, $\frac{\partial}{\partial \mu_n} L_k(v_1, \ldots, v_N, \lambda_1, \ldots, \lambda_N, \mu_1, \ldots, \mu_N) = 0$, where

$$L_k = T(v_1, \ldots, v_N) + \sum_{n=1}^{N} \left( \frac{1}{2} (1 - v_n^T v_n) \lambda_n \right) + \sum_{n=1}^{N} (g_n(v_n), \mu_n).$$

with, $g_n(v_n) := [\langle v_n, v_1^1 \rangle, \ldots, \langle v_n, v_n^{k-1} \rangle]^T \in \mathbb{R}^{k-1}$ To illustrate this procedure it is applied to the tensor of Example 3. The first basis vectors have to satisfy,

$$T(\cdot, y^{(1)}, z^{(1)}) = \lambda_1^{(1)} \langle \cdot, x^{(1)} \rangle,$$

$$T(x^{(1)}, \cdot, z^{(1)}) = \lambda_1^{(1)} \langle y^{(1)}, \cdot \rangle,$$

$$T(x^{(1)}, y^{(1)}, \cdot) = \lambda_1^{(1)} \langle z^{(1)}, \cdot \rangle.$$

The basis vectors are all on the unit ball, $|| \cdot || = 1$, the superscript (1) is dropped for the elements in the vectors,

$$x_1^2 + x_2^2 = 1, \quad y_1^2 + y_2^2 = 1, \quad z_1^2 + z_2^2 = 1.$$

The first basis vectors and their singular value is obtained,

$$\lambda = \frac{1}{2} + \frac{1}{2} \sqrt{5},$$

$$x^{(1)} = \left( \begin{array}{c} \frac{1}{\sqrt{2}} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{5}} \frac{1}{\sqrt{5}} \end{array} \right), \quad y^{(1)} = \left( \begin{array}{c} 1 \\ 0 \end{array} \right), \quad z^{(1)} = \left( \begin{array}{c} \frac{1}{\sqrt{2}} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{5}} \frac{1}{\sqrt{5}} \end{array} \right).$$

The second basis vectors values, $x^{(2)}, y^{(2)}, z^{(2)}$ have to satisfy the extra conditions,

$$\langle x^{(2)}, x^{(1)} \rangle = 0, \quad \langle y^{(2)}, y^{(1)} \rangle = 0, \quad \langle z^{(2)}, z^{(1)} \rangle = 0.$$

This gives,

$$\lambda^{(2)} = 0, \quad \mu_x = 0, \quad \mu_y = \frac{-\sqrt{5}}{2\frac{1}{2} + \frac{1}{2} \sqrt{5}}, \quad \mu_z = 0,$$

$$x^{(2)} = \left( \begin{array}{c} \pm \frac{1}{\sqrt{2}} \frac{1}{\sqrt{2}} \\ \pm \frac{1}{\sqrt{5}} \frac{1}{\sqrt{5}} \end{array} \right), \quad y^{(2)} = \left( \begin{array}{c} 0 \frac{1}{2} \frac{1}{2} \end{array} \right), \quad z^{(2)} = \left( \begin{array}{c} \pm \frac{1}{\sqrt{2}} \frac{1}{\sqrt{2}} \\ \pm \frac{1}{\sqrt{5}} \frac{1}{\sqrt{5}} \end{array} \right).$$

The tensor product of the first basis vectors found by the SVD have minimal error approximating the original, measured in Frobenius norm.

**Theorem 2.** Given a tensor $T$, let $\sigma_1$ be its first singular value, $u_1, \ldots, u_N$ the corresponding singular vectors. Let

and for $k > 1$,

$$\sigma_k(T) = \sup_{v_n \in S_n^k, 1 \leq n \leq N} \| T(v_1, \ldots, v_N) \|,$$

where,

$$S_n^k := \{ v \in V_n \| v \| = 1, \langle v, v_n^j \rangle = 0 \text{ for } j = 1, \ldots, (k - 1) \}.$$
Then \( \| T - T_1 \|_{Fro} \) is minimal among all rank-one approximations of \( T \), i.e. \( \| T - T_1 \|_{Fro} \leq \| T - T_0 \|_{Fro} \) \( \forall T_0 \in T_N \) of rank one.

The proof is given in Appendix A.

**Example 5.** The TSVD applied on the same data obtained in Example 4 gives a set of basis functions in \( x \), \( y \) and in \( t \).

In figure 5 are the first five basis functions in both the \( x \) and \( y \) direction shown.

### C. Successive Rank-1 decomposition

HOSVD and TSVD impose orthogonality conditions on the separate vector spaces the tensor operates on. From a physical viewpoint this is for example for the heat equation a strange condition. Apparently, the orientation of the coordinates gives different optimal approximations. Intuitively one would like to impose the orthogonality condition on the total tensor space.

Although orthogonality in all the separate vector spaces the tensor operates on implies orthogonality of the total tensor. Orthogonality of just one basis vector to the space spanned by its preceding basis vectors is enough to guarantee the rank one tensor to be orthogonal to the rank one tensors of the preceding basis vectors. This follows from,

\[
\langle u_1^{(n)} \otimes \cdots \otimes u_N^{(n)}, \sum_{i_1=1}^{n-1} \cdots \sum_{i_N=1}^{n-1} u_1^{(i_1)} \otimes \cdots \otimes u_N^{(i_N)} \rangle = 0,
\]

\[
\langle u_1^{(n)}, \sum_{i_1=1}^{n-1} \alpha_1^{(i)} \rangle \cdots \langle u_N^{(n)}, \sum_{i_N=1}^{n-1} \alpha_N^{(i)} \rangle = 0.
\]

The successive rank one tensor decomposition is defined as,

\[
T = \sum_{i=1}^{R} \sigma_i (u_1, \ldots, u_N)^{(i)} ,
\]

with,

\[
\langle (u_1, \ldots, u_N)^{(i)}, (u_1, \ldots, u_N)^{(j)} \rangle = \begin{cases} 0 & i \neq j \\ 1 & i = j \end{cases}.
\]

The rank one tensors \( U^{(i)} = \sigma_i (u_1, \ldots, u_N)^{(i)} \) are obtained from the optimisation problem,

\[
U^{(i)} = \arg \max_{\|U^{(i)}\|_1 = 1} \langle T, U^{(i)} \rangle.
\]

The values \( \sigma_i \) are determined by,

\[
\sigma_i = \langle T, U^{(i)} \rangle.
\]

This decomposition has some nice properties on the optimality of the error made by rank-\( R \) approximations. A big disadvantage though is that it is hard to obtain a ODE from a PDE by using these basis functions in the Galerkin projections, because the orthogonality of the separate basis functions is not a condition anymore.

### V. Simulation results

The HOSVD decomposition made in Example 4 and the TSVD decomposition on the same data are compared for their use in reduction. First the reduction of the solution data, by truncation of the expansion (5) will be reviewed. The method that uses the matrix SVD for the separation of signals in a spatial and a temporal signal will also be applied on this data.
so one can compare it with the tensor decompositions. The basis obtained from these decompositions will then be used to simulate the reduced dynamical systems, with again the same parameters used in Example 4.

A. Reduction of the solution

The tensor decomposition gives basis functions that can be used in the expansion, \( u_{(t, x, y)}(t, x, y) = \sum_{i}^{r_t} \sum_{j}^{r_y} \sum_{k}^{r_z} \tau(t) \xi^i(x) \eta^j(y) \), where \( r_x, r_y, r_t \) are the order of truncation in the different directions. Since only \( \{ \xi(x)^j \} \) and \( \{ \eta(y)^k \} \) will be used in the simulations of the reduced systems. The approximations of the solution by a truncated expansion of only truncations in the \( x \) and \( y \) direction are interesting to compare. Thus the basis \( \tau(t) \) will be taken full, \( u_{(t, x, y)}(t, x, y) = \sum_{i}^{r_t} \sum_{j}^{r_y} \sum_{k}^{r_z} \tau(t) \xi^i(x) \eta^j(y) \). Notice this is impossible for the normal POD method that uses the matrix SVD, since the number of spatial basis functions are equal to the number of temporal basis functions in the reduced expansion of the normalised maximal difference between the original and the approximation averaged over time thus, \( E(x, y) := \frac{1}{N_t} \sum_{i=1}^{N_t} \max_{x,y} \left| T(x, y, t_i) - A_{r_x, r_y}(x, y, t_i) \right| / \left\| T(x, y, t_i) \right\|_{Fro} \).

This measure for all of the rank \( r_x \times r_y \) approximations is plotted in Fig. 7 for the HOSVD, and in Fig. 8 for the TSVD, where \( r_x \) range through 1 to 61 and \( r_y \) through 1 to 81.

B. Reduced system simulations

The basis obtained from the HOSVD and the TSVD are used to simulate the set of ODEs, (8) that were obtained from the heat equation of Example 4. Again the Frobenius norm of the error is calculated for up to 5 basis functions in each spatial dimension. These are depicted in Table IV for the HOSVD and Table V for the TSVD.

Another measure to compare those methods is looking at the normalised maximal difference between the original and the approximation averaged over time thus, \( E(x, y) := \frac{1}{N_t} \sum_{i=1}^{N_t} \max_{x,y} \left| T(x, y, t_i) - A_{r_x, r_y}(x, y, t_i) \right| / \left\| T(x, y, t_i) \right\|_{Fro} \).

Table II shows these norms of the error for the HOSVD and Table III for the TSVD.

VI. NON-CARTESIAN DOMAINS

Data for the generation of the basis functions is usually obtained from Finite Element Method, FEM, simulations. Those methods are optimised to find a grid that will described the physics as accurately as possible on the least points needed. Often this results in a non equidistant and non square grid.

<table>
<thead>
<tr>
<th>Order</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>(</td>
<td></td>
<td>E_r</td>
<td></td>
<td>_{Fro} )</td>
<td>24.041</td>
</tr>
<tr>
<td>TABLE I: Error norm</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( r_x )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>( r_y )</td>
<td>28.486</td>
<td>24.100</td>
<td>22.963</td>
<td>22.251</td>
<td>22.072</td>
</tr>
<tr>
<td>2</td>
<td>27.991</td>
<td>18.124</td>
<td>15.231</td>
<td>13.378</td>
<td>12.914</td>
</tr>
<tr>
<td>3</td>
<td>27.245</td>
<td>16.701</td>
<td>13.216</td>
<td>11.002</td>
<td>10.066</td>
</tr>
<tr>
<td>4</td>
<td>27.011</td>
<td>15.933</td>
<td>11.932</td>
<td>9.196</td>
<td>7.232</td>
</tr>
<tr>
<td>5</td>
<td>26.965</td>
<td>15.814</td>
<td>11.733</td>
<td>8.848</td>
<td>6.269</td>
</tr>
<tr>
<td>TABLE II: Error norm basis function on ( r_x \times r_y ) approximation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( r_x )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>( r_y )</td>
<td>28.724</td>
<td>25.353</td>
<td>25.356</td>
<td>24.754</td>
<td>24.701</td>
</tr>
<tr>
<td>2</td>
<td>28.036</td>
<td>20.016</td>
<td>19.643</td>
<td>17.693</td>
<td>17.564</td>
</tr>
<tr>
<td>3</td>
<td>27.967</td>
<td>19.722</td>
<td>18.705</td>
<td>16.194</td>
<td>15.981</td>
</tr>
<tr>
<td>4</td>
<td>27.847</td>
<td>19.280</td>
<td>18.224</td>
<td>15.063</td>
<td>14.761</td>
</tr>
<tr>
<td>5</td>
<td>27.774</td>
<td>19.271</td>
<td>18.208</td>
<td>14.950</td>
<td>14.571</td>
</tr>
<tr>
<td>TABLE III: Error norm ( r_x \times r_y ) approximation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( r_y )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>( r_x )</td>
<td>28.761</td>
<td>25.830</td>
<td>25.662</td>
<td>25.977</td>
<td>26.466</td>
</tr>
<tr>
<td>5</td>
<td>28.366</td>
<td>20.424</td>
<td>19.413</td>
<td>18.113</td>
<td>18.547</td>
</tr>
<tr>
<td>TABLE IV: Error norm ( r_x \times r_y ) basis function on ( r_x \times r_y ) approximation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The structure of such a grid is called the mesh structure. When applying the normal SVD on such data this mesh structure can be translated into a gramian that weights the inner product such that the Euclidean inner product on the discrete function values represents the inner product of the interpolated functions.

As an example a flow problem on a non Cartesian domain, see Fig. 9 was simulated in Comsol a FEM solver. It simulated this on a mesh structure defined by triangulation. A snapshot of this is shown in Fig. 10.

Example 6 (Flow problem). The PDE of the flow problem that was simulated is given as,

$$\rho \left( \frac{\partial u}{\partial t} + (u \cdot \nabla) u \right) = \eta \Delta u - \nabla P,$$

$$\nabla \cdot u = 0.$$ 

A. Relation of the SVD for continuous functions with SVD on the discretised functions

All the projections and data manipulations were stated on continuous functions but the computation is done on discrete data. Therefore the numerical answers have to be related to the original continuous functions, this is interpolation.

One way to view sampling (discretisation) is to represent signals $u \in X$ on a finite dimensional basis, $\phi_1, \ldots, \phi_N$ of
elements in \( X \). That is, any sampled \( u \in \mathcal{U} \) can be written as,
\[
u = \sum_k u_k \phi^k.
\]

Call coefficients \( u_k \) sampling coefficients. Let \( \mathcal{U} = \text{span}\{\phi^1, \ldots, \phi^N\} \), then \( \dim \mathcal{U} = N, \mathcal{U} \subset X \)

**Definition 10** (Gramian). The Gramian \( G \in \mathbb{R}^{N \times N} \) is given by,
\[
G_{kl} = \langle \phi^k, \phi^l \rangle.
\]

Note that, \( G \) is symmetric and \( G \succ 0 \).

**Lemma 2.** The inner product of 2 arbitrary solutions can be inferred from sample coefficients. If \( u', u'' \) are elements in \( \mathcal{U} \), with sampling coefficients \( u_k', u_k'' \) respectively. Then,
\[
\langle u', u'' \rangle = u'^T G u'',
\]
where \( u' = \text{col}(u_1', \ldots, u_N') \), \( u'' = \text{col}(u_1'', \ldots, u_N'') \) denote the vectors of sampling coefficients of \( u' \) and \( u'' \), respectively.

Conversely, a partial POD basis of \( X \) can be obtained from sample coefficients as follows.

**Theorem 3.** Let \( U_{\text{snap}} = USV^T \) be an SVD of \( U_{\text{snap}} \). Let \( G^{-1/2} U = [u^1, \ldots, u^N] \) with \( u^n \in \mathbb{R}^N \). Define
\[
\xi_n := \sum_{k=1}^N u_k^n \phi_k,
\]
then \( \xi_n \in X \}, \{\xi_n\} \) is a POD basis for \( \mathcal{U} \).

**Proof:**
\[
\langle \xi_m, \xi_n \rangle = \bar{u}^T_m G \bar{u}_n = \bar{u}^T_m G^{-1/2} G G^{-1/2} u_n = \bar{u}^T_m u_n = \delta_{mn}.
\]

**B. Separation of functions on non Cartesian domains**

There are several ways of applying the treated methods to functions on non Cartesian domains, \( \Omega \subset \mathbb{R} \).

The first way would be to extend the values on the discretised non-Cartesian domain with arbitrary values such that the data is represented as an N-way array. The interpolation functions will not consider the artificial values. And by using the Gramian to weight the inner product the tensor decomposition will give little significance on the added data. This way one treats the inner product on the non Cartesian domain as a weighted inner product on the Cartesian domain, where the weights are allowed to become zero, thus not defining an inner product.

The second way would be to find a coordinate transformation, \( \pi : X \times Y \rightarrow \Omega \subset \mathbb{R}^2 \) such that Hilbert spaces on the Cartesian coordinates can be related to the original Hilbert space. Thus inner products can be calculated in the new separated coordinates. Functions on the new coordinate domain \( X \times Y \) can be expressed as sum of separable functions, \( f(x,y) = \sum_i f^i(x)f^i(y) \). By applying the coordinate transformation the inner product on \( \Omega \) can be expressed as an inner product on \( X \times Y \) as follows,
\[
\int_{\Omega} f(x,y) g(x,y) \, dx \, dy = \int_{X \times Y} f(\pi(x,y)) g(\pi(x,y)) \det(\nabla \otimes \pi) \, dx \, dy,
\]
where \( (\nabla \otimes \pi) \) represents the Jacobian of \( \pi \). Suppose \( \pi \) to have the property that \( p(x,y) := \det(\nabla \otimes \pi) \) is separable as \( p(x,y) = p(x)p(y) \) and substituting \( f(\pi(x,y)) = \sum_i f^i(x)f^i(y) \) and \( g(\pi(x,y)) = \sum_j g^j(x)g^j(y) \), one can express the inner product as,
\[
\langle f, g \rangle = \sum_{i,j} \int_X f^i(x)g^i(y)p(x)dx \int_Y f^j(y)g^j(y)p(y)dy = \sum_{i,j} \langle f^i, g^j \rangle_X \langle f^i, g^j \rangle_Y,
\]
where \( \langle \cdot, \cdot \rangle_X, \langle \cdot, \cdot \rangle_Y \) (are defined in the obvious way) are inner products if and only if \( p(x) > 0 \) on \( X \), \( p(y) > 0 \) on \( Y \).

An alternative third way would be to introduce a (nonlinear) coordinate transformation, only without the previous derived conditions. This amounts to computing the inner product on \( \Omega \) only, separation is in \( X \) and \( Y \). In Fig. 11 an assignment of new coordinates for the problem of Example 6 is shown, such that it is possible to separate the function on the new domain.

![Fig. 11: Coordinate transformation](image)

**VII. Symmetry**

A big disadvantage of the POD method is that its effectiveness is strongly related to the coordinates in which the data is represented, to illustrate this see Example 7.

**Example 7** (Traveling pulse). Let \( u(t,x) \) be the solution of the PDE,
\[
\frac{\partial u}{\partial t} = \nu \frac{\partial^2 u}{\partial x^2},
\]
\( u(0,x) = u_0(x) \).

This simple PDE has the analytic solution, \( u(t,x) = u_0(x + \nu t) \). Now if \( u_0(x) \) is taken to be a smooth pulse like a \( \sin(\frac{x}{a}) \) function, discretised such that the snapshot data at time intervals collected into a matrix, results in a identity matrix. This means that, the singular vectors are equally important. This means although the dynamics are very simple, the POD method is not able to find a better basis. Changing
the coordinates gives the system,
\[ \dot{w} = \nu, \]
\[ u(t, x) = u(x - w). \]

Thus the same dynamics can be simulated using just one basis function on different coordinates. This change of coordinates gives an exact representation in which the PDE is changed to an ODE without model reduction. Since the dynamics it describes is still the same the search for a reduction technique that does not depend on the coordinate system is started. A possible way to achieve this is by describing the dynamics in a coordinate free way. Operations like the gradient, divergence, rotation, Laplacian are all independent on the coordinate system that is used to express them. In other words applying the operation in one coordinate system or doing a coordinate transformation first then apply the operation and do a coordinate transformation back to the original coordinates has the same results. A way to describe these operations without using coordinates is by using differential forms.[9][10]

The afore mentioned operations are defined in 3-dimensional space. They can also be extended to n-dimensional spaces by using the exterior derivative, the exterior product and the Hodge operator. Some literature [11][12][13] deals with finding symmetries in PDEs by using these operators from differential geometry.

This procedure, first introduced in [11], creates from the PDE a set of differential forms such that when the dependent variables are considered to be functions of the independent variables and the forms are set equal to zero, called annulling they form the original PDE. Annulling restricts the forms to the manifold of the solution of the PDE. Now the method looks for invariances of the PDE, this means the equations are still satisfied after a transformation. By solving for the invariance of the Lie derivative of these forms, the determining equations for the symmetry generators are obtained.

This method shows some promise for doing reduction in a coordinate free way, yet there are still no practical results made.

VIII. Conclusion

In this project the use of tensor decompositions in model reduction applications was studied. More specifically three types of reduction strategies have been compared, each of them based on a different spectral decomposition of signals. For the reduction of PDEs these POD by tensor decomposition methods still have some difficulties. Compared to the normal POD they achieve a lesser accurate reduction in number of basis functions. The HOSVD seems work better than the TSVD, yet it is hard to make a valid comparison, because this could be numerical implementation issues for the TSVD. This algorithm is still in development, while the HOSVD makes use of the well developed SVD algorithm for matrices. It should be noted that the conclusions on performance are based on the application of the methods to an example. The successive rank one approximations are not useful for obtaining a set of ODEs from a PDE by Galerkin projections, yet one could change the method to one that imposes the condition for time basis functions to be orthonormal. Applying the Galerking projection on those basis functions could result in a ODE.

Also a major concern should be the application of these methods to functions on non Cartesian domains and how boundary conditions get translated to these reduced models. As those boundary conditions determine most of the behaviour a PDE describes.

Another application for these tensor decompositions is identification on multi dimensional data. For example data obtained from batch processes that will exhibit dynamics in time and in batch number. These tensor decompositions could lead to multi dimensional controller design, where the process variation in batch number is also controlled. The value of separating the basis functions is more clear for these kind processes than the processes described by PDEs.

The real computational reduction will be from the reduction of the number of ODEs that have to be solved. Typically for linear problems the ODE can be written in a state space representation. The \( A \) matrix does not have to be full rank, so one could try to reduce this set of equation to a smaller set. For the simple dynamics of the heat equation it was observed it is possible to separate the dynamics into two sets of ODEs, one for each direction.

For the dynamics describe by PDEs the reduction by finding symmetries would be an good extension to the reduction by POD.

Acknowledgment

I would like to thank Siep Weiland, Femke van Belzen, Jobert Ludlage and Leila Ózkan for their supervision. The people from IPCOS to give me the opportunity to perform my research at their company. And my fellow graduate students for invariances of the PDE, this means the equations are still satisfied after a transformation. By solving for the invariance of the Lie derivative of these forms, the determining equations for the symmetry generators are obtained.

This method shows some promise for doing reduction in a coordinate free way, yet there are still no practical results made.

A. Proofs

Lemma 2
\[ \langle (f(x, y), g(x))_X, h(y)_Y \rangle = \int_X \int_Y f(x, y)g(x)dxh(y)dy, \]
\[ = \int_X \int_Y f(x, y)g(x)h(y)dxdy, \]
\[ = \int_X \int_Y f(x, y)g(x)h(y)dydx, \]
\[ = \int_X \int_Y f(x, y)h(y)dyg(x)dx, \]
\[ = \langle (f(x, y), h(y))_Y, g(x)_X \rangle. \]

\[ \square \]

Theorem 2
Given a tensor \( T \), a rank one approximation, \( T_1 = \sigma_1(u_1 \otimes \cdots \otimes u_N) \) where \( u_i \in S = \{ u_i \mid \| u_i \| = 1, (u_1 \otimes \cdots \otimes u_N) \} \), and \( \sigma_1 \) is obtained on \( \langle T, (u_1 \otimes \cdots \otimes u_N) \rangle \). The error norm, Frobenius norm, is
minimized by the vectors \( \{ u_i \} \) obtained from the maximisation problem, 
\[
\arg \min_{u_1, \ldots, u_N} \| T(u_1, \ldots, u_N) \| 
\]
\[
= \arg \min_{u} \| T - T_1 \| 
\]
\[
= \arg \min_{u \in \mathcal{S}} \langle T - T_1, T - T_1 \rangle 
\]
\[
= \arg \min_{u \in \mathcal{S}} (\langle T, T \rangle - 2 \langle T, T_1 \rangle + \langle T_1, T_1 \rangle) 
\]
\[
= \arg \min_{u \in \mathcal{S}} \begin{pmatrix} \langle T, T \rangle - 2 \langle T, \sigma_1(u_1 \otimes \cdots \otimes u_N) \rangle \\
+ |\langle \sigma_1(u_1 \otimes \cdots \otimes u_N), \sigma_1(u_1 \otimes \cdots \otimes u_N) \rangle | \\
\end{pmatrix} 
\]
\[
= \min_{u \in \mathcal{S}} \begin{pmatrix} \langle T, T \rangle - 2 \langle T, \sigma_1(u_1 \otimes \cdots \otimes u_N) \rangle \\
+ \sigma_1^2 \langle u_1 \otimes \cdots \otimes u_N, u_1 \otimes \cdots \otimes u_N \rangle \\
\end{pmatrix} 
\]
\[
= \min_{u \in \mathcal{S}} \begin{pmatrix} \langle T, T \rangle - 2 \langle T, \sigma_1(u_1 \otimes \cdots \otimes u_N) \rangle \\
+ \sigma_1^2 \langle u_1 \otimes \cdots \otimes u_N, u_1 \otimes \cdots \otimes u_N \rangle + \sigma_1^2 \\
\end{pmatrix} 
\]
\[
= \min_{u \in \mathcal{S}} \begin{pmatrix} \langle T, T \rangle - 2 \langle T, (u_1 \otimes \cdots \otimes u_N) \rangle \\
+ \langle (u_1 \otimes \cdots \otimes u_N), (u_1 \otimes \cdots \otimes u_N) \rangle \\
\end{pmatrix} 
\]
\[
= \min_{u \in \mathcal{S}} \begin{pmatrix} \langle T, T \rangle - 2 \langle T, (u_1 \otimes \cdots \otimes u_N) \rangle \\
+ \langle (u_1 \otimes \cdots \otimes u_N), (u_1 \otimes \cdots \otimes u_N) \rangle \end{pmatrix} 
\]
\[
= \min_{u \in \mathcal{S}} \begin{pmatrix} \langle T, (u_1 \otimes \cdots \otimes u_N) \rangle \\
+ \langle (u_1 \otimes \cdots \otimes u_N), (u_1 \otimes \cdots \otimes u_N) \rangle \end{pmatrix} 
\]
\[
= \min_{u \in \mathcal{S}} \begin{pmatrix} \langle T, T \rangle - 2 \langle T, (u_1 \otimes \cdots \otimes u_N) \rangle \\
+ \langle (u_1 \otimes \cdots \otimes u_N), (u_1 \otimes \cdots \otimes u_N) \rangle \end{pmatrix} 
\]
\[
= \min_{u \in \mathcal{S}} \begin{pmatrix} \langle T(u_1, \ldots, u_N) \rangle \\
+ \langle (u_1 \otimes \cdots \otimes u_N), (u_1 \otimes \cdots \otimes u_N) \rangle \end{pmatrix} 
\]
\[
\text{Lemma 3. The set } \{ e_x^i \otimes e_y^j \} \text{ is a basis for } \mathcal{H} = \mathcal{X} \otimes \mathcal{Y}, \\
\zeta = \sum_{i,j} \xi_i e_x^i \otimes e_y^j. \\
\text{Proof:} \\
\mathcal{X} \otimes \mathcal{Y} \ni \zeta(\cdot, \cdot) : \mathcal{X} \times \mathcal{Y} \to \mathbb{R} \\
\zeta(\xi, \eta) = \xi(\sum_i \eta_i e_x^i, \sum_j \eta_j e_y^j) \\
\text{By the application of multi linearity of the elements of } \mathcal{X} \otimes \mathcal{Y} \\
\text{this is,} \\
\sum_{i,j} \langle e_x^i, \eta \rangle \xi(\cdot, e_y^j) = \sum_{i,j} \sum \langle e_x^i, \eta \rangle \xi(\cdot, e_y^j) \\
\sum_{i,j} \langle e_x^i, \xi(\cdot, e_y^j) \eta \rangle = \sum_{i,j} \sum \langle e_x^i, \xi(\cdot, e_y^j) \eta \rangle \\
\text{Thus,} \\
\zeta = \sum_{i,j} \xi(\cdot, \eta) \langle e_x^i, \otimes e_y^j \rangle. \\
\]"