CADI AYYAD UNIVERSITY
FACULTY OF SCIENCE AND TECHNOLOGY MARRAKECH

Master in Modelling and Scientific Calculations For Mathematical Engineering (MOCASIM)

Final year project

Iterative Algorithms for Large Linear and Multilinear Dynamical Systems

Realized by :

ELHASSANI Ayoub

Supervisors :

Pr. SADEK El Mostafa
Pr. KREIT Karim

Examiners :

Pr. ALAA Nour Eddine	FSTG, UCA, Marrakech.
Pr. BENTBIB H. Abdeslem	FSTG, UCA, Marrakech.
Pr. BENCHETTOU Oumaima	UFR SEGMI , UPN, Paris.
Pr. KREIT Karim	FSTG, UCA, Marrakech.
Pr. SADEK El Mostafa	ENSA, UCD, El Jadida.

ANNÉE UNIVERSITAIRE : 2023/2024

ACKNOWLEDGMENT

In the name of Allah, the Most Gracious, the Most Merciful.
"My Lord, increase me in knowledge" (Ta-Ha, 114)

With a grateful heart, I start by thanking Allah, who has been my guide, my companion, and the source of all good in my life. His wisdom has given me patience, strength, and perseverance, reminding us that "Indeed, Allah is with the patient" (Al-Baqarah, 153).

I'm very grateful to my supervisors, Mr. Sadek El Mostapha and Mr. Kreit Karim. Your patience, guidance, and motivation have been key in shaping this project and my academic growth. Your wisdom and support mean a lot to me. Thank you to Mr. Abdeslem Hafid Bentbib for his generous help and direction. I'm very thankful to Mr. Nour Eddine Alaa, both as my teacher and a jury member, for his valuable motivation and for evaluate this project. I sincerely thank Ms. Benchettou Oumaima for agreeing to be on my jury and for evaluating my project.

A special thanks to the brilliant PhD students Warda Abdelmouhsin, Hicham Zourak, Ali Jad, and Boubekraoui Maryam. Your support and teamwork have been crucial in completing this final year project.

My deepest appreciation goes to the pillars of my life my beloved family. To my cherished mother, Aicha, and my esteemed father, My Ahmed, your unconditional love and support have been my foundation. To my dear sisters, Smahane and Yassmine, and my brother Mohamed, your encouragement has been my strength. My heartfelt thanks extend to every member of my family.

Table des matières

Chapitre 1

Introduction

In the erea of Big Data, we have access to an enormous amount of data from experiments and real-time recordings. This data can be utilized for optimization, control, and monitoring of complex industrial models. A key challenge we face is how to sift through this vast amount of information to extract the critical data needed for specific tasks. Additionally, we may ask if our decision-making can be enhanced by integrating this data with physics-based models.

Applied mathematics has long been used to efficiently design dynamic models through sophisticated software and digital tools. These models help us understand various phenomena and physical processes. Today, with the advent of modern, powerful supercomputers equipped with extensive memory and high-speed operations, we can use them for diverse tasks such as weather forecasting, scientific research, and intricate simulations. However, when computational resources are limited or when execution time is critical, new challenges arise, necessitating the development of new strategies.

A highly accurate dynamical model is often preferred to represent physical systems or phenomena because it leads to a better understanding, whether through simulation, optimization, control, or analysis. However, the complexity of these models can vary depending on the desired level of accuracy. Generally, higher accuracy results in higher complexity, making the models more difficult to manage and sometimes impractical for engineers. Simulations can take days or even weeks, which is not feasible for systems with

limited computational power, such as those in cars, airplanes, or air conditioners.
One strategy to address this issue, which avoids solving the full model, is model approximation. This approach involves a set of methods aimed at reducing the complexity of models. Model approximation seeks efficient techniques to construct simpler, reduced dynamical models that still accurately replicate the original input-output relationships and retain the key characteristics.

Dynamical models are essential tools that can be applied in various fields, from physical systems like weather patterns, fluid flow, and celestial mechanics, to biological systems such as population dynamics and disease spread, and social systems like economic models and network dynamics. These models help simulate complex systems, predict their behavior under different conditions, and design interventions for control or optimization.

In many industrial applications, it is practical to build a single high-fidelity model and then simplify it based on the specific application and utilization. This leads to the concept of model order reduction. The goal of this method is to develop effective approaches to create high-fidelity, low-complexity dynamical models. These models are cheaper to execute while maintaining certain characteristics of the original model, such as stability, accuracy, and structure.

In model order reduction methods, there are both intrusive and non-intrusive approaches. The intrusive approach is used when we have a complete description of the model, meaning the system operators are available. Conversely, the non-intrusive approaches, which are data-based methods, assume that the underlying abstract model structure is unknown. Both methods aim to produce a reduced model that is easier to manipulate and control.

In general many physical phenomena are modeled by PDEs. The discretization of these equations often leads to dynamical systems (continuous or discrete) depending on a control vector whose choice can stabilize the dynamical system.
In our study, we will focus on projection methods, specifically developing and analyzing techniques based on Krylov processes for model reduction in large-scale Multi-Input Multi-Output (MIMO) linear time-invariant dynamical systems.

Chapitre 2

Linear dynamical systems

2.1 Introduction to linear dynamical systems

2.1.1 Preliminaries in linear systems theory

A continuous linear dynamical system can be expressed in the form of an algebraic differential equation :

$$\begin{cases} \dot{x}(t) = A(t)x(t) + B(t)u(t), \\ y(t) = C(t)x(t) + D(t)u(t), \end{cases} \tag{2.1}$$

when the matrix coefficients (A(t), B(t), C(t), D(t)) in (2.1) do not depend on time or do not vary much over periods of time, then they can be replaced with constant coefficients, which gives a time-invariant dynamical system ("Linear Time-Invariant" LTI)

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t), \\ y(t) = Cx(t) + Du(t), \end{cases} \tag{2.2}$$

where $A \in \mathbb{R}^{n\times n}, B \in \mathbb{R}^{n\times p}, C^T \in \mathbb{R}^{n\times s}$ and $D \in \mathbb{R}^{s\times p}$, with $p, s \ll n$. $x(t) \in \mathbb{R}^n$ denotes the state vector and it belongs to the state space, $u(t) \in \mathbb{R}^p$ is the input (or the control) vector and $y(t) \in \mathbb{R}^s$ is the output (to be measured). A are supposed to be large and sparse, B and C are respectively input and output matrices.

If $s = p = 1$, then the LTI dynamical system (2.2) is called Single-Input Single-Output (SISO) and is called Multiple-Input Multiple-Output (MIMO) other-

wise. The control problem consists of acting on the input vector $u(t)$ so that the output vector $y(t)$ has a desirable time trajectory and modifying the input $u(t)$ according to the output $y(t)$ which is observed or to the state $x(t)$ is called feedback.
The LTI dynamical system (2.2) can also be denoted as

$$\mathbf{\Sigma} \equiv \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$$

This last description of the linear system is called the internal description, it uses both the input vector u(t) and the state vector x(t) for the output vector y(t).
Another characterization of the linear system under the name of external description can be written as follows.

$$y(t) = h \star u := \int_{-\infty}^{+\infty} h(t-\tau)u(\tau)d\tau$$

where $h(t)$ "Kernel" or "weighting pattern" of Σ. The function $h(t)$ is called "impulse response" when $u(t) = \delta(t)$, the Dirac Delta function, and in this case, $y(t) = h(t)$. Note that any LTI system can be represented by a convolution with a suitable choice of $h(t)$. This last description only involves the input vector u(t) for the output vector y(t) and this through the function $h(t)$, which obviously depends on the matrix coefficients (A, B, C, D). For example, the "impulse response" function of the stable LTI system (2.2) is given by :

$$h(t) = \begin{cases} C\exp(At)B + D\delta(t), & t \geq 0 \\ 0, & t < 0 \end{cases}$$

Definition 2.1.1.

The representation $\Sigma = (A, B, C, D)$ is known as the realization of the system (2.2). We define the order of (2.2) as the dimension of the corresponding state-space "n".
A realization with the smallest dimension is called minimal. It means that the dynamical system is as economical as possible in terms of the number of independent parameters needed to describe the state of the system.

Note that the realization of a system is not unique in the sense of its input-output behavior.

2.1.2 Transfer function

An important measurement to study the characteristic of the LTI system is the transfer function. It is also known as the Input-Output representation, which is an external representation obtained by eliminating the state vector between the state and output equations with zero initial conditions. In order to get this function, we apply the Laplace transform,

$$\mathcal{L}f(s) = \int_0^\infty f(t)e^{-st}dt.$$

to the state equation (2.2), and we get

$$\begin{cases} sX(s) = AX(s) + BU(s) \\ Y(s) = CX(s) + DU(s) \end{cases}$$

where $X(s) = \mathcal{L}(x(t))$, $Y(s) = \mathcal{L}(y(t))$ and $U(s) = \mathcal{L}(u(t))$. Therefore

$$X(s) = (sI - A)^{-1}BU(s)$$

and by substituting $X(s)$ in the output equation of (2.2), we get

$$Y(s) = F(s)U(s), \tag{2.3}$$

with

$$F(s) = C(sI - A)^{-1}B + D. \tag{2.4}$$

The rational function $F(s)$ is called the transfer function of the system (2.2). This transfer function connects the input and output by $Y(s) = F(s)U(s)$ in the frequency domain. It is reminded that most model reduction techniques are based on this transfer function.

Definition 2.1.2.

Two LTI (Linear Time-Invariant) systems $\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$ *and* $\left[\begin{array}{c|c} \widetilde{A} & \widetilde{B} \\ \hline \widetilde{C} & D \end{array}\right]$ *are called equivalent if they have the same transfer function, i.e.,*

$$\tilde{F}(s) = \tilde{C}(sI_n - \tilde{A})^{-1}\tilde{B} + \tilde{D} = C(sI_n - A)^{-1}B + D = F(s).$$

It is easy to verify that for any non-singular $n \times n$ matrix T, the LTI system

$$\left[\begin{array}{c|c} T^{-1}AT & T^{-1}B \\ \hline CT & 0 \end{array}\right]$$

is equivalent to the LTI system $\left[\begin{array}{c|c} A & B \\ \hline C & 0 \end{array}\right]$. The interest in defining the system in the frequency domain is to obtain several equivalent systems which give us the freedom to choose the most stable version. Under this transformation, the states are connected by the relation $x(t) = T\widetilde{x}(t)$.
The resolution of (2.2) with an initial condition $x_0 = x(t_0)$ give

$$x(t) = \exp(A(t - t_0))x_0 + \int_{t_0}^{t} \exp(A(t - \tau))Bu(\tau)d\tau.$$

Thus, the output vector $y(t)$ in the time domain is written as

$$y(t) = C\exp(A(t - t_0))x_0 + C\int_{t_0}^{t} \exp(A(t - \tau))Bu(\tau)d\tau.$$

Comparing this last one with that in the frequency domain

$$Y(s) = F(s)U(s) = (C(sI_n - A)^{-1}B)U(s),$$

we see that the expression in the frequency domain is indeed much simpler.

2.1.3 Moments of a transfer function

One of the methods used in the reduction of large-scale models are the methods based on moments and/or Markov parameters.
The transfer function can be expanded into a Taylor series around a point $s_0 \in \mathbb{C}$, giving :

$$F(s) = \eta_0(s_0) - \eta_1(s_0)(s - s_0) + \ldots + (-1)^j \eta_j(s_0)(s - s_0)^j + \ldots \tag{2.5}$$

such that

$$\eta_j(s_0) = C(s_0 I_n - A)^{-(j+1)} B \quad \forall j \geq 0. \tag{2.6}$$

The matrix coefficient $\eta_j(s_0)$ is called the j-th moment of the system at s_0 for $j \geq 0$.
In the special case where $s_0 = \infty$, the transfer function F can be expanded into a Laurent series as follows :

$$F(s) = \frac{1}{s} C(I_n - A)^{-1} B s = \frac{1}{s} \sum_{i=0}^{\infty} \eta_i(\infty) s^{-i} \tag{2.7}$$

with $\eta_i(\infty) = CA^i B$.
In this case the matrix coefficients $\eta_i(\infty)$ are called : Markov parameters of F.
Many model reduction methods focus on the moment problem, particularly projection methods on Krylov spaces widely use this approach. The main question in the moment problem is to construct a reduced-order model (A_m, B_m, C_m) such that the first l moments of the reduced transfer function

$$F_m(s) = \eta_0(s_0) - \eta_1(s_0)(s - s_0) + \ldots + (-1)^j \eta_j(s_0)(s - s_0)^j + \ldots$$

and those of F coincide, i.e.,

$$\eta_j(s_0) = \eta_j(s_0) \quad j = 0 \ldots l - 1.$$

The same question applies to Markov parameters, i.e., $s_0 = \infty$.

2.1.4 Basic system properties

Stability, controllability, and observability

The controllability of a dynamical system is related to the ability of the system to attain a given state under the action of an appropriate control signal. If a state is not controllable, then it is not possible to move this state to another one by acting on the control input. If the matrix representing the dynamics of a non controllable state is stable, then the state is said to be stabilizable. The observability is related to the possibility of evaluating the state of a system through output measurements.
We notice that in the expression of the transfer function, the matrix D does not play a significant role. Hereafter, we set $D = 0$. Thus, the LTI system is written as :

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t). \end{cases} \tag{2.8}$$

Definition 2.1.3. ***(Stability)***

A matrix A is said to be stable if $\Lambda(A) \subset \mathbb{C}_-$, i.e., the real parts of all eigenvalues of A are negative. An LTI system is said to be stable if its matrix A is stable.
Where $\mathbb{C}_- := \{s \in \mathbb{C}; real(s) < 0\}$.

Definition 2.1.4.

A matrix-valued function $F(s) \in \mathbb{C}^{n\times n}$ (for $s \in \mathbb{C}$) is said to be real positive if the following three conditions are satisfied :

1. *All elements of $F(s)$ are analytic for $s \in \mathbb{C}_+$.*
2. *$F(s) \in \mathbb{R}^{n\times n}$ for $(s > 0)$.*
3. *$F(s) + F^*(s) \geq 0$ for $s \in \mathbb{C}_+$.*

Where $\mathbb{C}_+ := \{s \in \mathbb{C}; real(s) > 0\}$.

Definition 2.1.5. ***Passivity***

A stable LTI system is said to be passive if its transfer function is positive real, i.e., $F(s) + F^*(s) \geq 0$ for $s \in \mathbb{C}_+$.

Definition 2.1.6. ***Controllability***

A system is said to be controllable if from an initial null state, any state can be reached via suitable control, i.e., given $z \in \mathbb{R}^n$, if $x(t_0) = 0$, then there exists $u(t)$ such that $x(t) = z$.

A necessary and sufficient condition for controllability is given by :

Proposition 2.1.1. *A Linear Time-Invariant (LTI) system defined by $\dot{x}(t) = Ax(t) + Bu(t)$ is controllable if and only if*

$$\textit{rank}([B\, AB\, A^2B\, \ldots\, A^{n-1}B]) = n.$$

Definition 2.1.7. ***Observability***

An LTI system is said to be observable when, for $u(t) = 0$, $y(t)$ is uniquely determined by $x(t_0)$.

Proposition 2.1.2. *An LTI system is observable if and only if*

$$\textit{rank} \begin{pmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{pmatrix} = n.$$

Controllability and Observability Gramians

We assume that the LTI dynamic system is stable.

Definition 2.1.8.

The Gramian of controllability associated with the LTI system is defined by

$$P = \int_0^\infty e^{tA} B B^T e^{tA^T}\, dt, \tag{2.9}$$

and the Gramian of observability is defined by

$$Q = \int_0^\infty e^{tA^T} C^T C e^{tA}\, dt. \tag{2.10}$$

By applying the Parseval's identity to these last relations, we obtain the new

expressions for the Gramians :

$$P = \int_0^\infty (j\omega I - A)^{-1} B B^T (j\omega I - A^T)^{-1} \, d\omega, \tag{2.11}$$

$$Q = \int_0^\infty (j\omega I - A^T)^{-1} C^T C (j\omega I - A)^{-1} \, d\omega. \tag{2.12}$$

2.1.5 Different dynamical system norms

System norms play an important role in the analysis of LTI systems, as they quantify certain properties of the model and also they are used to measure the accuracy of the reduced order model. We will concentrate on the norm H_∞, other norms like the H_2 norm will be also introduced in this section ; see [4] for more details.

The H_2 norm

We start by recalling the definition of the H_2-norm of the transfer function $F(s)$ associated with the dynamical system Σ.

Definition 2.1.9.

The H_2-norm of $F(s)$ is defined as

$$\|F(\cdot)\|_{H_2}^2 = \frac{1}{2\pi} \int_{-\infty}^{+\infty} \textit{trace}(F(j\omega)^T F(j\omega)) d\omega,$$

where j is the complex number $j^2 = -1$.

Consider the impulse response $g(t) = \mathcal{L}^{-1}[F(s)] = Ce^{tA}B$ where $\mathcal{L}$ is the Laplace transform

$$\mathcal{L}(g)(s) = \int_0^\infty g(t) e^{-st} dt = F(s).$$

Then using the Parseval relation

$$\int_0^\infty \text{trace}(g(t)^T g(t)) dt = \frac{1}{2\pi} \int_{-\infty}^{+\infty} \text{trace}(F(j\omega)^T F(j\omega)) d\omega,$$

the H_2 norm can also be expressed as

$$\|F(\cdot)\|_{H_2}^2 = \int_0^\infty \text{trace}(g(t)^T g(t)) dt.$$

Therefore, the H_2 norm could be calculated as follows

$$\|F(\cdot)\|_{H_2}^2 = \textbf{trace}\left[B^T\left(\int_0^\infty e^{tA^T}C^TCe^{tA}\right)B\right].$$

Setting

$$Q = \int_0^\infty e^{tA^T}C^TCe^{tA}dt,$$

we get

$$\|F(\cdot)\|_{H_2}^2 = \textbf{trace}(B^TQB).$$

Assuming that A is a stable matrix, the observability Gramian Q can be computed by solving the Lyapunov matrix equation (2.13). We notice that in a similar way, the H_2 norm can be computed using the controllability Gramian defined by (2.9). Therefore, H_2 norm can be expressed as

$$\|F(\cdot)\|^2 = \textbf{trace}(CPC^T).$$

The H_∞-norm

Another important norm in linear system theory is the well-known H_∞-norm, which is related to the Hardy space $\mathcal{H}_\infty$.

Definition 2.1.10.

The H_∞ norm of the transfer function $F(\cdot)$ is defined as

$$\|F(\cdot)\|_{H_\infty} = \sup_{\omega\in\mathbb{R}} \sigma_{max}(F(j\omega)),$$

where σ_{max} denotes the largest singular value.

To approximate the H_∞-norm, we choose a set of frequencies $\Omega_n = \{\omega_1, \omega_2, \ldots, \omega_n\}$ and search for

$$\sup_{1\le k\le n} \sigma_{\max}(F(j\omega_k)) \approx \|F(\cdot)\|_{H_\infty}.$$

2.2 Global Arnoldi method for large MIMO dynamical system

2.2.1 Preliminaries

In this section, we review some notations and definitions which are used throughout this chapter.

For two matrices X and Y in $\mathbb{R}^{n\times p}$, we define the Frobenius inner product $\langle X, Y\rangle_F = \mathrm{Tr}\left(X^\top Y\right)$ where $\mathrm{Tr}\left(X^\top Y\right)$ denotes the trace of the square matrix $X^\top Y$. The associated Frobenius norm is given by $\|Y\|_F = \mathrm{Tr}\left(Y^T Y\right)^{\frac{1}{2}}$.

A sequence $V_1, V_2, \ldots, V_m$ of elements of $\mathbb{R}^{n\times p}$ is said to be F-orthonormal if it is orthonormal with respect to the inner product $\langle .\ ,\ . \rangle_F$, i.e., $\langle V_i, V_j\rangle_F = \delta_{i,j}$.

For $Y \in \mathbb{R}^{n\times p}$, we denote by $\mathrm{vec}(Y)$ the vector of $\mathbb{R}^{np}$ obtained by stocking the columns of Y. For two matrices X and $Y, X\otimes Y = [x_{i,j}Y]$ denotes the Kronecker product of the matrices X and Y. In the sequel, we give some properties of the Kronecker product.

1. $(A\otimes B)^\top = A^\top \otimes B^\top$.
2. $(A\otimes B)(C\otimes D) = (AC\otimes BD)$.
3. If A and B are non singular matrices of size $n\times n$ and $p\times p$ respectively, then the $np\times np$ matrix $A\otimes B$ is non singular and $(A\otimes B)^{-1} = A^{-1}\otimes B^{-1}$.
4. $\mathrm{vec}(A)^\top \mathrm{vec}(B) = \mathrm{Tr}\left(A^\top B\right)$.

Definition 2.2.1.

Let $A = [A_1, \ldots, A_s]$ and $B = [B_1, \ldots, B_l]$ be matrices of dimension $n\times$ sp and $n\times lp$, respectively, where A_i and B_j are in $\mathbb{R}^{n\times p}$. Then the $s\times l$ matrix $A^\top \diamond B$ is defined by :

$$A^\top \diamond B = \left[\langle A_i, B_j\rangle_F\right]_{1\le i\le s; 1\le j\le l}.$$

Remark 2.2.1.

1. *The matrix $A = [A_1, \ldots, A_s]$ is F-orthonormal if and only if $A^\top \diamond A = I_s$.*
2. *For all $X \in \mathbb{R}^{n\times p}$, we have $X^\top \diamond X = \|X\|_F^2$.*
3. $(DA)^\top \diamond B = A^\top \diamond \left(D^\top B\right)$.

4. $A^\top \diamond (B\,(L \otimes I_p)) = \left(A^\top \diamond B\right) L$.

5. $\left\|A^\top \diamond B\right\|_F \leq \|A\|_F \|B\|_F$.

Now, we proceed to build a process similar to QR to calculate an $n \times ks$ F-orthonormal matrix $Q = [Q_1, Q_2, \ldots, Q_k]$ such that $Q^T \diamond Q = I_k$. This process is known by The Global QR factorization which is summarized in the following :

Algorithm 1 Global QR Algorithm

Inputs : $Z = [Z_1, Z_2, \ldots, Z_k]$, an $n \times sk$ matrix
Outputs : $Q = [Q_1, Q_2, \ldots, Q_k]$, an $n \times sk$ F-orthonormal matrix; R a $k \times k$ upper triangular matrix
Initialize $r_{1,1} = \|Z_1\|_F$
Set $Q_1 = Z_1 / r_{1,1}$
for $j = 2, \ldots, k$ **do**
  $W = Z_j$
  for $i = 1, \ldots, j-1$ **do**
    $r_{i,j} = \mathrm{trace}(W^T Q_i)$
    $W = W - r_{i,j} Q_i$
  end for
  $r_{j,j} = \|W\|_F$
  if $r_{j,j} = 0$ **then**
    stop
  end if
  $Q_j = W / r_{j,j}$
end for

2.2.2 Global Arnoldi algorithm

The global Krylov subspace $\mathcal{K}_m(A, V)$ is the subspace of $\mathbb{R}^{n \times p}$ generated by the matrices $V, AV, \ldots, A^{m-1}V$,

$$\mathcal{K}_m(A, V) = \mathrm{span}\{V, AV, \ldots, A^{m-1}V\}. \tag{2.13}$$

The global Arnoldi algorithm computes an F-orthonormal basis $\{V_1, \ldots, V_m\}$ of the matrix Krylov subspace $\mathcal{K}_m(A, V)$ satisfying

$$(V_i, V_j)_F = \delta_{ij}, \quad i, j = 1, \ldots, m \tag{2.14}$$

where $\delta_{ij} = 0$ if $i \neq j$ and 1 if $i = j$.

We now describe the global Arnoldi algorithm for computing an F-orthonormal basis of the global Krylov subspace (2.14).

Algorithm 2 The modified global Arnoldi algorithm

Compute the $n \times p$ matrix V_1 such that $V_1 = V/\|V\|_F$.
for $j = 1, \ldots, m$ **do**
    $\tilde{V} = AV_j$,
    for $i = 1, \ldots, j$ **do**
        $h_{i,j} = \mathbf{trace}(\tilde{V}^T V_i)$,
        $\tilde{V} = \tilde{V} - h_{i,j} V_i$,
    end for
    $h_{j+1,j} = \|\tilde{V}\|_F$,
    $V_{j+1} = \tilde{V}/h_{j+1,j}$.
end for

Setting $\mathcal{V}_m = [V_1, \ldots, V_m]$, and using the $\diamond$ product, the F-orthogonality relation (2.13) could be written as

$$\mathcal{V}_m^T \diamond \mathcal{V}_m = I.$$

Add to an F-orthonormal basis $\{V_1, \ldots, V_m\}$ of the global Krylov subspace, the global Arnoldi process produces an upper Hessenberg matrix $\tilde{H}_m = [h_{i,j}] \in \mathbb{R}^{(m+1)\times m}$, $i = 1, \ldots, m+1$ and $j = 1, \ldots, m$ and an upper Hessenberg matrix H_m obtained by deleting the last column of $\tilde{H}_m$.

Proposition 2.2.1. *Suppose that m steps of the algorithm have been computed. Then we have the following projection property :*

$$A\mathcal{V}_m = \mathcal{V}_m \diamond \tilde{H}_m \tag{2.15}$$

$$= \mathcal{V}_m \diamond H_m + h_{m+1,m} V_{m+1} E_m^T label e7 \tag{2.16}$$

Where $E_m^T = [0_p, \cdots, 0_p, I_p]$ *and* $\mathcal{V}_{\mathrm{m}}^T \diamond (A\mathcal{V}_{\mathrm{m}}) = \mathrm{H_m}$.

2.2.3 Application to MIMO systems

In this section, we consider the linear time invariant (LTI) system defined in (2.8)

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases}$$

where $A \in \mathbb{R}^{n\times n}, B \in \mathbb{R}^{n\times p}$, and $C \in \mathbb{R}^{s\times n}$. For the sake of simplicity, we can assume that $x(0) = 0$.

As mentioned above the classical way of relating the input to output is to use a transfer function (or impulse response in the time domain) of the previous LTI system.

$$F(z) = C\left(zI_n - A\right)^{-1} B.$$

We recall that most of model reduction techniques, like the moment-matching approaches, are based on this transfer function. If the number of state variables is very high, it is not practical to use the full system for simulation or runon-time control. So it is reasonable to look for models of low order that approximate the behavior of the original models. The low-order model is expressed as follows :

$$\begin{cases} \dot{\hat{x}}(t) = \hat{A}\hat{x}(t) + \widehat{B}u(t) \\ \hat{y}(t) = \widehat{C}\hat{x}(t) \end{cases} \tag{2.17}$$

where $\hat{A} \in \mathbb{R}^{k\times k}, \widehat{B} \in \mathbb{R}^{k\times p}$ and $\widehat{C} \in \mathbb{R}^{s\times k}$ with $k \ll n$.

Now, let us see how to obtain a reduced order model by using the global Arnoldi process. This will be done by approximating the transfer function (2.17).

Let us write $F(z)$ as $F(z) = CX$ where $X \in \mathbb{R}^{n\times p}$ is the solution of the following matrix linear system :

$$\left(zI_n - A\right) X = B$$

In order to approximate the transfer function $F(z)$, we seek for an approximate solution X_m to X via the global Full Orthogonalization Method (Gl-FOM).

Starting from an initial guess $X_0 = 0$ and applying the global FOM to the previous system, we can show that the approximate solution X_m is given by

$$X_m = \beta \mathcal{V}_m \left[\left(zI_m - H_m\right)^{-1} e_1 \otimes I_s\right]$$

where $\mathcal{V}_m$ and H_m represent the matrix Krylov basis and the Hessenberg matrix obtained by applying the global Arnoldi process to the pair (A, B). Hence, $F(z)$ is approximated by

$$\begin{aligned} F_m(z) = CX_m &= \beta C\mathcal{V}_m \left[(zI_m - H_m)^{-1} e_1 \otimes I_s\right] = \beta C\mathcal{V}_m \left[(zI_m - H_m)^{-1} \otimes I_s\right](e_1 \otimes I_s) \\ &= \beta C\mathcal{V}_m \left[(zI_m - H_m) \otimes I_s\right]^{-1}(e_1 \otimes I_s) = \beta C\mathcal{V}_m \left[(zI_m \otimes I_s - H_m \otimes I_s)\right]^{-1}(e_1 \otimes I_s) \\ &= C\mathcal{V}_m \left[zI_{ms} - (H_m \otimes I_s)\right]^{-1} \beta\,(e_1 \otimes I_s)\,. \end{aligned}$$

This suggests that the reduced order model can be given by

$$\begin{cases} \dot{x}_m(t) = A_m x_m(t) + B_m u(t) \\ y_m(t) = C_m x_m(t) \end{cases}$$

where $A_m = (H_m \otimes I_s) \in \mathbb{R}^{ms\times ms}, B_m = \beta\,(e_1 \otimes I_s) \in \mathbb{R}^{ms\times s}$ and $C_m = C\mathcal{V}_m \in \mathbb{R}^{r\times ms}$. Moreover, the transfer function of the previous reduced order model is given by

$$F_m(z) = C_m\,(zI_{ms} - A_m)^{-1} B_m$$

The above approach for obtaining a reduced order model is summarized in the following algorithm.

Model reduction via the global Arnoldi process

— Inputs : A the system matrix, B the input matrix, C the output matrix, m the index of the reduced-order model.

— Step 1. Apply m steps of Algorithm 1 to the pair (A, B) to get the matrices $\mathcal{V}_m$ and H_m.

— Step 2. Compute $A_m = (H_m \otimes I_s)\,, B_m = \beta\,(e_1 \otimes I_s)$ and $C_m = C\mathcal{V}_m$.

Next, we show that the above reduced order model approximates the behavior of the original model. More precisely, we have the following results.

Theoreme 2.2.1. *[1] The matrices A_m, B_m and C_m generated by applying the global Arnoldi process to (2.13) are such that the first m Markov parameters of the original and the reduced models are the same, that is,*

$$CA^jB = C_m A_m^j B_m, \quad \textit{for } j = 0, 1, \ldots, m-1.$$

Proof 2.2.1. *Let $\mathcal{V}_m = [V_1, \ldots, V_m]$ and H_m be, respectively, the matrix Krylov basis and the Hessenberg matrix generated by applying m steps of the global Arnoldi process to the pair (A, B). Then, it is not difficult to show that for any polynomial ϕ of degree less than m, there holds :*

$$\phi(A)\mathcal{V}_m \left(e_1 \otimes I_s\right) = \mathcal{V}_m \left(\phi\left(H_m\right) e_1 \otimes I_s\right)$$

Hence, for any $j \in \{0, \ldots, m-1\}$ and using the fact that $B = \beta V_1 = \beta \mathcal{V}_m \left(e_1 \otimes I_s\right)$ we have

$$CA^j B = \beta C A^j \mathcal{V}_m \left(e_1 \otimes I_s\right) = \beta C \mathcal{V}_m \left(H_m^j e_1 \otimes I_s\right)$$

Finally, as $\left(H_m^j e_1 \otimes I_s\right) = \left(H_m^j \otimes I_s\right)\left(e_1 \otimes I_s\right)$ for $j = 0, 1, \ldots, m-1$, we obtain

$$CA^j B = C\mathcal{V}_m \left(H_m^j \otimes I_s\right) \beta \left(e_1 \otimes I_s\right) = C_m A_m^j B_m$$

which ends the proof.

In the following result we give an upper bound for the norm of the error $F(z) - F_m(z)$ that can be used as a stopping criterion.

Theoreme 2.2.2. *If the matrices $zI_n - A$ and $zI_{ms} - (H_m \otimes I_s)$ are invertible, then*

$$\left\|F(z) - F_m(z)\right\|_2 \leqslant h_{m+1,m} \|C\|_2 \left\|\left(zI_n - A\right)^{-1}\right\|_2 \left\|V_{m+1}\right\|_2 \left\|\Gamma_m(z)\right\|_2, \tag{2.18}$$

where $\Gamma_m(z) = \left(e_m^{\mathrm{T}} \otimes I_s\right)\left(zI_{ms} - (H_m \otimes I_s)\right)^{-1}\left(e_1 \otimes I_s\right)$. Furthermore, for $|z| > \|A\|_2$, we have

$$\left\|F(z) - F_m(z)\right\|_2 \leqslant h_{m+1,m} \frac{\|C\|_2 \left\|V_{m+1}\right\|_2 \left\|\Gamma_m(z)\right\|_2}{|z| - \|A\|_2}. \tag{2.19}$$

Proof 2.2.2. *It is easy to see that*

$$\left(zI_n - A\right)\mathcal{V}_m = \mathcal{V}_m \left(zI_{ms} - H_m \otimes I_s\right) - h_{m+1,m} V_{m+1} E_m^{\mathrm{T}}$$

So, using the fact that $B = \beta V_1 = \beta \mathcal{V}_m E_1$, where $E_1 = e_1 \otimes I_s$, we have

$$\begin{aligned} F(z) - F_m(z) &= C\left(zI_n - A\right)^{-1} B - C_m\left(zI_n - A_m\right)^{-1} B_m \\ &= C\left(zI_n - A\right)^{-1}\left[B - \beta\left(zI_n - A\right)\mathcal{V}_m\left(zI_{ms} - H_m \otimes I_s\right)^{-1} E_1\right] \\ &= C\left(zI_n - A\right)^{-1}\left[B - \left(\beta\mathcal{V}_m E_1 - h_{m+1,m} V_{m+1} E_m^{\mathrm{T}}\left(zI_{ms} - H_m \otimes I_s\right)^{-1} E_1\right)\right] \\ &= h_{m+1,m} C\left(zI_n - A\right)^{-1} V_{m+1} E_m^{\mathrm{T}}\left(zI_{ms} - H_m \otimes I_s\right)^{-1} E_1 \end{aligned}$$

Letting $\Gamma_m(z) = E_m^{\mathrm{T}}\left(zI_{ms} - (H_m \otimes I_s)\right)^{-1} E_1$ *and using the inequality*

$$\left\|(zI_n - A)^{-1}\right\|_2 \leqslant \frac{1}{|z| - \|A\|_2},$$

we obtain (2.18) and (2.19) for $|z| > \|A\|_2$.

2.3 Block Arnoldi method for large MIMO dynamical system

The block Krylov subspace $\mathbb{K}_m^b(A, V)$ is a subspace of $\mathbb{R}^n$ generated by the columns of the matrices $V, AV, \ldots, A^{m-1}V$

$$\mathbb{K}_m^b(A, V) = \mathrm{Range}\left(\left[V, AV, \ldots, A^{m-1}V\right]\right).$$

The block Arnoldi algorithm generates a sequence of blocks $\{V_1, \ldots, V_m\}$ of size $n \times p$ such that their columns form an orthonormal basis of the block Krylov subspace $\mathcal{K}_m(A, V)$ defined above. The algorithm is defined as follows.

After m steps we get the following results :

• The real matrix of size $n \times mp$ defined by :

$$\mathbb{V}_m = [V_1, \cdots, V_m] \text{ is orthogonal : } \mathbb{V}_m^T \mathbb{V}_m = I_{mp}.$$

• The real matrix $\mathbb{H}_m$ of size $mp \times mp$, whose non-zero blocks H_{ij} of size $p \times p$ are given by the previous algorithm.

$$H_m = (H_{ij})_{1 \leq i,j \leq m} \quad H_{ij} \equiv 0, \text{ for } i > j + 1 \text{ is of Hessenberg type.}$$

Algorithm 3 The modified block Arnoldi algorithm

Require: $A \in \mathbb{R}^{n\times n}$, $V \in \mathbb{R}^{n\times p}$, m.
  Compute $[V_1, \Lambda] = QR(V), \mathbb{V}_1 = [V_1]$.
  for $j = 1, \ldots, m$ **do**
    $\hat{V}_{j+1} = AV_j$.
    Orthogonalize $\hat{V}_{j+1}$ with respect to $[\mathbb{V}_1, \ldots, \mathbb{V}_j]$ to get V_{j+1}, i.e.,
    for $i = 1, 2, \ldots, j$ **do**
      $H_{i,j} = (V_i)^T \hat{V}_{j+1}$;
      $\hat{V}_{j+1} = \hat{V}_{j+1} - V_i H_{i,j}$;
    end for
    $[V_{j+1}, H_{j+1,j}] = QR(\hat{V}_{j+1})$.
    $\mathbb{V}_{j+1} = [\mathbb{V}_j, V_{j+1}]$.
  end for

• Let E_m be the real matrix of size $mp \times p$ composed of the last p columns of the identity matrix I_{mp},

$$E_m^T = [0_p, \cdots, 0_p, I_p] .$$

• We have the following projection property :

$$A\mathbb{V}_m = \mathbb{V}_m \mathbb{H}_m + V_{m+1} H_{m+1,m} E_m^\top \tag{2.20}$$

2.3.1 Application to MIMO sytems

Similarly to the global case, we consider the linear time invariant (LTI) system

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases}$$

where $A \in \mathbb{R}^{n\times n}, B \in \mathbb{R}^{n\times p}$, and $C \in \mathbb{R}^{s\times n}$, and assuming that $x(0) = 0$.

As we mentioned in the global case the classical way of relating the input to output is to use a transfer function (or impulse response in the time domain) givev by :

$$F(z) = C\left(zI_n - A\right)^{-1} B. \tag{2.21}$$

We look for models of low order that approximate the behavior of the original models. The low-order model is expressed as follows :

$$\begin{cases} \dot{\hat{x}}(t) = \hat{A}\hat{x}(t) + \widehat{B}u(t) \\ \hat{y}(t) = \widehat{C}\hat{x}(t) \end{cases}$$

where $\hat{A} \in \mathbb{R}^{k\times k}, \widehat{B} \in \mathbb{R}^{k\times p}$ and $\widehat{C} \in \mathbb{R}^{s\times k}$ with $k \ll n$.

Now, using the block Arnoldi process describe by the algorithm above let us see how to obtain a reduced model. This will be done by approximating the transfer function.

Let us consider $F(z)$ as $F(z) = CX$ where $X \in \mathbb{R}^{n\times p}$ is the solution of the following linear system :

$$(zI_n - A)\, X = B$$

Applying block Arnoldi process to the pairs (A, B) we get the matrix $\mathbb{V}_m$ and an upper block Hessenberg matrix $\mathbb{H}_m$. Considering the approximation $X \approx \mathbb{V}_m Y_m$, and using the Galerkin condition, we get the projected linear problem

$$\mathbb{V}_m^T(sI_n - A)\mathbb{V}_m Y_m = \mathbb{V}_m^T B,$$

which is equivalent to

$$(zI_{ms} - \mathbb{H}_m)Y_m = B_m.$$

Then we get the approximate transfer function

$$F_m(z) = C\mathbb{V}_m = C_m(zI - \mathbb{H}_m)^{-1}B_m, \quad \mathbb{H}_m = \mathbb{V}_m^T A\mathbb{V}_m, \quad C_m = C\mathbb{V}_m.$$

This suggests that the reduced order model can be given by

$$\begin{cases} \dot{x}_m(t) = A_m x_m(t) + B_m u(t) \\ y_m(t) = C_m x_m(t) \end{cases}$$

where $A_m = \mathbb{H}_m \in \mathbb{R}^{ms\times ms}, B_m = \mathbb{V}_m^T B \in \mathbb{R}^{ms\times s}$ and $C_m = C\mathbb{V}_m \in \mathbb{R}^{r\times ms}$.
The above approach for obtaining a reduced order model is summarized in the following algorithm.

Model reduction via the block Arnoldi process

— Inputs : A the system matrix, B the input matrix, C the output matrix,

m the index of the reduced-order model.

— Step 1. Apply m steps of Algorithm 2 to the pair (A, B) to get the matrices $\mathbb{V}_m$ and $\mathbb{H}_m$.

— Step 2. Compute $A_m = \mathbb{H}_m, B_m = \mathbb{V}_m^T B$ and $C_m = C\mathbb{V}_m$.

2.4 Extended Arnoldi method for large MIMO dynamical system

Given the matrix $A \in \mathbb{R}^{n\times n}$ is nonsingular and v is a vector in $\mathbb{R}^n$, with a fixed integer m, we define the classical extended Arnoldi Krylov subspace $K_m(A, v)$ as the space spanned by the vectors $A^{-m}v$, $A^{-2}v$, up to $A^{m-1}v$. A more recent development includes a convergence analysis for this subspace, providing new general estimates for the convergence rate for nonsingular matrices A. For a matrix V in $\mathbb{R}^{n\times p}$, the extended block Krylov subspace $K_m(A, V)$ is defined by the span of the columns of matrices $A^k V$, for k ranging from $-m$ to $m-1$.

The subspace $\mathcal{K}_m(A, V)$ is denoted by

$$\mathcal{K}_m(A, V) = \text{Range}\{A^{-m}V, \ldots, A^{-2}V, A^{-1}V, V, AV, A^2V, \ldots, A^{m-1}V\}.$$

The subspace $\mathcal{K}_m(A, V)$ is the sum of the simple extended Krylov subspaces $\mathcal{K}_m(A, V^{(i)}), i = 1, \ldots, r$ where $V^{(i)}$ is the i-th column of the matrix V. Notice that $Z \in \mathcal{K}_m(A, V)$ means that

$$Z = \sum_{i=-m}^{m-1} A^i V \Omega_i, \quad \text{where } \Omega_i \in \mathbb{R}^{p\times p}, i = -m, \ldots, m-1.$$

On the other hand, the extended matrix or global Krylov subspace $\mathcal{K}_m(A, V) \subseteq \mathbb{R}^{n\times p}$ is the subspace of matrices in $\mathbb{R}^{n\times p}$ spanned by $A^k V$, $k = -m, \ldots, m-1$, i.e.,

$$\mathcal{K}_m(A, V) = \text{span}\{A^{-m}V, \ldots, A^{-2}V, A^{-1}V, V, AV, A^2V, \ldots, A^{m-1}V\},$$

and hence $Z \in \mathcal{K}_m(A, V)$ iff $Z = \sum_{i=-m}^{m-1} \alpha_i A^i V$, $\alpha_i \in \mathbb{R}$.

We will consider two cases for the extended Arnoldi method : the block version

and the global version.

2.4.1 Block case

The extended block Arnoldi algorithm generates a sequence of blocks $\{V_1, \ldots, V_m\}$ of size $n \times 2r$ such that their columns form an orthonormal basis of the extended block Krylov subspace $\mathcal{K}_m(A, V)$. The algorithm is defined as follows :

Algorithm 4 The extended block Arnoldi algorithm

Require: $A \in \mathbb{R}^{n\times n}$, $V \in \mathbb{R}^{n\times r}$, m.
Compute $[V_1, \Lambda] = QR([V, A^{-1}V])$, $\mathbb{V}_1 = [V_1]$.
for $j = 1, \ldots, m$ **do**
    Set V_j^1 : first r columns of V_j ; V_j^2 : second r columns of V_j.
    $\hat{V}_{j+1} = [AV_j^1, A^{-1}V_j^2]$.
    Orthogonalize $\hat{V}_{j+1}$ with respect to $[\mathbb{V}_1, \ldots, \mathbb{V}_j]$ to get V_{j+1}, i.e.,
    for $i = 1, 2, \ldots, j$ **do**
        $H_{i,j} = (V_i)^T \hat{V}_{j+1}$;
        $\hat{V}_{j+1} = \hat{V}_{j+1} - V_i H_{i,j}$;
    end for
    $[V_{j+1}, H_{j+1,j}] = QR(\hat{V}_{j+1})$.
    $\mathbb{V}_{j+1} = [\mathbb{V}_j, V_{j+1}]$.
end for

Since Algorithm 4 implicitly involves a Gram-Schmidt procedure, the obtained block vectors $\mathbb{V}_m = [V_1, V_2, \ldots, V_m]$ ($V_i \in \mathbb{R}^{n\times 2r}$) have their columns mutually orthogonal provided that none of the upper triangular matrices $H_{j+1,j}$ are rank deficient.
Hence, after m steps, Algorithm 4 builds an orthonormal basis $\mathbb{V}_m$ of the extended block Krylov subspace $\mathcal{K}_m(A, V)$ and a block upper Hessenberg matrix $\mathbb{H}_m$ whose non zeros blocks are the $H_{i,j} \in \mathbb{R}^{2r\times 2r}$.

Let $\mathbb{T}_m = [T_{ij}] \in \mathbb{R}^{2mr\times 2mr}$ be the restriction of the matrix A to the extended Krylov subspace $\mathcal{K}_m(A, V)$, i.e.,

$$\mathbb{T}_m = \mathbb{V}_m^T A \mathbb{V}_m.$$

Proposition 2.4.1. *$\mathbb{T}_m$ is an upper block Hessenberg with $2r \times 2r$ blocks.*

Proof 2.4.1. *For any $m \geq 1$, we have*

$$A\mathcal{K}_m \subseteq \mathcal{K}_{m+1}. \tag{2.22}$$

then for $k > j+1$, $j = 1, 2, \ldots$ $V_k^T A V_j = 0$, thus T_m is block upper Hessenberg.

Proposition 2.4.2. *Let $\tilde{\mathbb{T}}_m = \mathbb{V}_{m+1}^T A \mathbb{V}_m$, and suppose that m steps of Algorithm 2 have been carried out. Then we have*

$$A\mathbb{V}_m = \mathbb{V}_{m+1}\tilde{\mathbb{T}}_m \tag{2.23}$$

$$= \mathbb{V}_m \mathbb{T}_m + V_{m+1} T_{m+1,m} E_m^T. \tag{2.24}$$

where $T_{i,j}$ is the $2s \times 2s$ (i,j) block of $\mathbb{T}_m$ and $E_m = \left[0_{2s\times 2(m-1)s}, I_{2s}\right]^T$ is the matrix of the last $2s$ columns of the $2ms \times 2ms$ identity matrix I_{2ms}.

Proof 2.4.2. *Since $\mathbb{V}_{m+1} = [\mathbb{V}_m, V_{m+1}]$, we have*

$$\mathbb{T}_{m+1} = \mathbb{V}_{m+1}^T A \mathbb{V}_{m+1}$$

$$= \begin{bmatrix} \mathbb{V}_m^T A \mathbb{V}_m & \mathbb{V}_m^T A V_{m+1} \\ V_{m+1}^T A \mathbb{V}_m & V_{m+1}^T A V_{m+1} \end{bmatrix}$$

$$= \begin{bmatrix} \mathbb{T}_m & \mathbb{V}_m^T A V_{m+1} \\ V_{m+1}^T A \mathbb{V}_m & V_{m+1}^T A V_{m+1} \end{bmatrix}.$$

Now, as $\mathbb{T}_{m+1}$ is block upper Hessenberg, we have $V_{m+1}^T A \mathbb{V}_m = T_{m+1,m} E_m^T$, and

$$\tilde{\mathbb{T}}_m = \mathbb{V}_{m+1}^T A \mathbb{V}_m = \begin{bmatrix} \mathbb{T}_m \\ T_{m+1,m} E_m^T \end{bmatrix}.$$

Using the fact that $A\mathcal{K}_m \subseteq \mathcal{K}_{m+1}$ and that $\mathbb{V}_{m+1}$ is orthogonal, it follows that there exists a matrix L such that $A\mathbb{V}_m = \mathbb{V}_{m+1} L$. Hence $\mathbb{V}_{m+1}^T A \mathbb{V}_m = L$, which shows that $L = \tilde{\mathbb{T}}_m$. Therefore, $A\mathbb{V}_m = \mathbb{V}_{m+1}\tilde{\mathbb{T}}_m$.

Now, suppose that m iterations of Algorithm 4 have been executed without encountering any breakdowns and that the upper block Hessenberg matrices

$\tilde{\mathbb{H}}_m$ and $\tilde{\mathbb{T}}_m$ are partitioned under the form :

$$\tilde{\mathbb{H}}_m = \left[h_{:,1}, h_{:,2}, \ldots, h_{:,2m}\right], \quad \tilde{\mathbb{T}}_m = \left[t_{:,1}, t_{:,2}, \ldots, t_{:,2m}\right]$$

where $h_{:,j} = \tilde{\mathbb{H}}_m \tilde{e}_j \in \mathbb{R}^{2(m+1)r \times r}$ and $t_{:,j} = \tilde{\mathbb{T}}_m \tilde{e}_j \in \mathbb{R}^{2(m+1)r \times r}$ with $\tilde{e}_j = e_j \otimes I_r$ and $\otimes$ denotes the Kronecker product. Moreover, suppose that the parameters Λ in line 1 and the block entry $H_{j+1,j}$ in line 11 of Algorithm 4 are such that

$$\Lambda = \begin{bmatrix} A_{1,1} & A_{1,2} \\ 0 & A_{2,2} \end{bmatrix}, \quad H_{j+1,j} = \begin{bmatrix} H_{j+1,j}^{1,1} & H_{j+1,j}^{1,2} \\ 0 & H_{j+1,j}^{2,2} \end{bmatrix},$$

where $A_{i,k}$, $H_{j+1,j}^{i,k}$ are $\mathbb{R}^{r \times r}$.

The computation of $\mathbb{T}_m$ seems to require additional matrix-vector products with A . We next in the following propositions derive a recursion for $\mathbb{T}_m$ that completely avoids this expensive step.

Proposition 2.4.3. *Let $\tilde{\mathbb{T}}_m$ and $\tilde{\mathbb{H}}_m$ be the upper block Hessenberg matrices as defined earlier. Then, for $j = 1, \ldots, m$, the odd block columns are such that*

$$t_{:,2j-1} = h_{:,2j-1}, \tag{2.25}$$

while the even columns satisfy

$$t_{:,2} = \left[\tilde{e}_1 \Lambda_{1,1} - t_{:,1}\Lambda_{1,2}\right] (\Lambda_{2,2})^{-1}, \tag{2.26}$$

$$t_{:,2j+2} = \left[\tilde{e}_{2j} - t_{:,1:2j+1} h_{1:2j+1,2j}\right] \left(H_{j+1,j}^{2,2}\right)^{-1}. \tag{2.27}$$

Proof 2.4.3. *From lines 7–11 of Algorithm 4 and since $V_i = [V_i^1, V_i^2]$ for $i = 1, \ldots, j+1$, we check that*

$$V_{j+1} H_{j+1,j} = \left[AV_j^1, A^{-1}V_j^2\right] - \mathbb{V}_j \mathbb{H}_j [\tilde{e}_{2j-1}, \tilde{e}_{2j}], \tag{2.28}$$

and so

$$\begin{aligned} AV_j^1 &= \mathbb{V}_j \mathbb{H}_j \tilde{e}_{2j-1} + V_{j+1}^{1,1} H_{j+1,j}^{1,1} \\ &= [\mathbb{V}_j, V_{j+1}^1] \begin{bmatrix} \mathbb{H}_j \tilde{e}_{2j-1} \\ H_{j+1,j}^{1,1} \end{bmatrix} \\ &= \mathbb{V}_{m+1} \tilde{\mathbb{H}}_m \tilde{e}_{2j-1}. \end{aligned}$$

Since $t_{:,2j-1} = \mathbb{V}_{m+1}^T A V_j^1$*, we get (2.25) by pre-multiplying the previous equality by* $\mathbb{V}_{m+1}^T$.
From the QR decomposition in line 1 of Algorithm 4, we have :

$$\begin{cases} V = V_1^1 \Lambda_{1,1}, \\ A^{-1} V = V_1^1 \Lambda_{1,2} + V_1^2 \Lambda_{2,2}. \end{cases}$$

Thus, by multiplying the second equality on the left by A*, we get*

$$\begin{aligned} A V_1^2 \Lambda_{2,2} &= V - A V_1^1 \Lambda_{1,2} \\ &= V_1^1 \Lambda_{1,1} - A V_1^1 \Lambda_{1,2}. \end{aligned}$$

As $t_{:,2} = \mathbb{V}_{m+1}^T A V_1^2$*, we get (2.26) by multiplying the above equality on the left by* $\mathbb{V}_{m+1}^T$.
From (2.28), we have

$$\begin{aligned} A^{-1} V_j^2 &= \mathbb{V}_j \mathbb{H}_j \tilde{e}_{2j} + V_{j+1} H_{j+1,j} \tilde{e}_2 \\ &= \mathbb{V}_j \mathbb{H}_j \tilde{e}_{2j} + V_{j+1}^1 H_{j+1,j}^{1,2} + V_{j+1}^2 H_{j+1,j}^{2,2}. \end{aligned}$$

This gives

$$\begin{aligned} V_{j+1}^2 H_{j+1,j}^{2,2} &= A^{-1} V_j^2 - \mathbb{V}_j \mathbb{H}_j \tilde{e}_{2j} - V_{j+1}^1 H_{j+1,j}^{1,2} \\ &= A^{-1} V_j^2 - \left[\mathbb{V}_j, V_{j+1}^1\right] \begin{bmatrix} \mathbb{H}_j \tilde{e}_{2j} \\ H_{j+1,j}^{1,2} \end{bmatrix}. \end{aligned}$$

Finally, relation (2.27) is obtained by right-multiplying with $\mathbb{V}_{m+1}^T A$ *and left-multiplying with* $(H_{j+1,j}^{2,1})^{-1}$ *the last equality. This completes the proof.*

In the next we will give some algebraic relations with the matrix A^{-1}, these relations could be used in moment matching techniques for model reduction in large scale Multiple Input Multiple Output (MIMO) dynamical systems. As we did above we will also consider the restriction of the matrix A^{-1} to the extended Krylov subspace $\mathcal{K}_m(A, V)$, i.e.,

$$\mathbb{L}_m = \mathbb{V}_m^T A^{-1} \mathbb{V}_m.$$

Notice that the matrix $\mathbb{L}_m = [L_{i,j}]$ is also an upper block Hessenberg matrix.

Proposition 2.4.4. *Assume that m steps of Algorithm 4 have been run and let $\overline{\mathbb{L}}_m = \mathbb{V}_{m+1}^T A^{-1} \mathbb{V}_m$, then we have the following relations*

$$\begin{aligned} A^{-1}\mathbb{V}_m &= \mathbb{V}_{m+1}\overline{\mathbb{L}}_m \\ &= \mathbb{V}_m \mathbb{L}_m + V_{m+1} L_{m+1,m} \mathbb{E}_m^T. \end{aligned} \tag{2.29}$$

Proof 2.4.4. *As $\mathbb{V}_{m+1} = [\mathbb{V}_m, V_{m+1}]$, we have*

$$\begin{aligned} \mathbb{L}_{m+1} &= \mathbb{V}_{m+1}^T A^{-1} \mathbb{V}_{m+1} \\ &= \begin{bmatrix} \mathbb{V}_m^T A^{-1} \mathbb{V}_m & \mathbb{V}_m^T A^{-1} V_{m+1} \\ V_{m+1}^T A^{-1} \mathbb{V}_m & V_{m+1}^T A^{-1} V_{m+1} \end{bmatrix} \\ &= \begin{bmatrix} \mathbb{L}_m & \mathbb{V}_m^T A^{-1} V_{m+1} \\ V_{m+1}^T A^{-1} \mathbb{V}_m & V_{m+1}^T A^{-1} V_{m+1} \end{bmatrix}. \end{aligned}$$

Now, since $\mathbb{L}_{m+1}$ is an upper block Hessenberg matrix, we also have

$$\left(V_{m+1}\right)^T A^{-1} \mathbb{V}_m = L_{m+1,m}^b \mathbb{E}_m^T,$$

and so the upper block Hessenberg matrix $\overline{\mathbb{L}}_m$ can be expressed as

$$\overline{\mathbb{L}}_m = \begin{bmatrix} \mathbb{L}_m \\ L_{m+1,m} \mathbb{E}_m^T \end{bmatrix}.$$

$\mathbb{V}_{m+1}$ is orthogonal and using the fact that $A^{-1}\mathbb{K}_m(A, V) \subseteq \mathbb{K}_{m+1}(A, V)$, there exists an upper block Hessenberg matrix L such that $A^{-1}\mathbb{V}_m = \mathbb{V}_{m+1} L$.

Then, $\mathbb{V}_{m+1}^T A^{-1}\mathbb{V}_m = L$*, which shows that* $L = \overline{\mathbb{L}}_m$*. Hence, we obtain*

$$A^{-1}\mathbb{V}_m = \mathbb{V}_{m+1}L = \mathbb{V}_{m+1}\overline{\mathbb{L}}_m = \mathbb{V}_m\mathbb{L}_m + V_{m+1}L_{m+1,m}\mathbb{E}_m^T,$$

which completes the proof.

We have $\mathbb{L}_m = [L_{i,j}]$. Moreover, the sub-matrices $L_{i+1,i} \in \mathbb{R}^{2r\times 2r}$ are such that the r first columns are zero. Hence, $L_{m+1,m}$ is partitioned under the form

$$L_{m+1,m} = \begin{bmatrix} 0_r & L_{m+1,m}^{1,2} \\ 0_r & L_{m+1,m}^{2,2} \end{bmatrix}.$$

and as $V_{m+1} = \left[V_{m+1}^1, V_{m+1}^2\right]$, the relation () becomes

$$A^{-1}\mathbb{V}_m = \mathbb{V}_m\mathbb{L}_m + \left[O_{n\times(2m-1)r}, V_{m+1}^1 L_{m+1,m}^{1,2} + V_{m+1}^2 L_{m+1,m}^{2,2}\right]$$

Next, we show how to compute the columns of the matrix $\overline{\mathbb{L}}_m$ without using A^{-1}.

The following result enables us to compute $\overline{\mathbb{L}}_m$ directly from the columns of the upper block Hessenberg matrix $\overline{\overline{H}}_m$ obtained from Algorithm 4.

Proposition 2.4.5. *Let* $\overline{\mathbb{L}}_m$ *and* $\overline{\mathbb{H}}_m$ *be the upper block Hessenberg matrices defined earlier. Then we have the following relations*

$$\overline{\mathbb{L}}_m\widetilde{e}_1 = \left[\widetilde{e}_1\Lambda_{1,2} + \widetilde{e}_1\Lambda_{2,2}\right](\Lambda_{1,1})^{-1} \tag{2.30}$$

and for $k = 1, \ldots, m$

$$\overline{\mathbb{L}}_m\widetilde{e}_{2k} = \overline{\mathbb{H}}_m\widetilde{e}_{2k} \tag{2.31}$$

and

$$\mathbb{L}_m\widetilde{e}_{2k+1} = \left(\widetilde{e}_{2k-1} - \begin{bmatrix} \overline{\mathbb{L}}_k \\ 0_{2(m-k)r\times 2kr} \end{bmatrix}\mathbb{H}_k\widetilde{e}_{2k-1}\right)\left(H_{k+1,k}^{1,1}\right)^{-1} \tag{2.32}$$

where $\widetilde{e}_i = e_i \otimes I_r$ *and the* e_i*'s are the vectors of the canonical basis.*

Proof 2.4.5. • *Let* $[V, A^{-1}V] = V_1^b\Lambda$ *be the QR decomposition of* $[V, A^{-1}V]$

which can be written as

$$[V, A^{-1}V] = V_1\Lambda = \begin{bmatrix} V_1^1 & V_1^2 \end{bmatrix} \begin{bmatrix} \Lambda_{1,1} & \Lambda_{1,2} \\ 0 & \Lambda_{2,2} \end{bmatrix}. \tag{2.33}$$

• *For* $k = 1, \ldots, m$*, let's partition the lower triangular matrix* $H_{k+1,k}$ *under the form*

$$H_{k+1,k} = \begin{bmatrix} H_{k+1,k}^{1,1} & H_{k+1,k}^{1,2} \\ 0 & H_{k+1,k}^{2,2} \end{bmatrix}.$$

To prove (2.30), we start from the **QR** *decomposition of* $[V, A^{-1}V]$ *given in (2.33) :*

$$\left[V, A^{-1}V\right] = \left[V_1^1\Lambda_{1,1}, V_1^1\Lambda_{1,2} + V_1^2\Lambda_{2,2}\right]$$

Then, if $\Lambda_{1,1}$ *is nonsingular, we obtain*

$$A^{-1}V_1^1 = A^{-1}V\Lambda_{1,1}^{-1} = \left[V_1^1\Lambda_{1,2} + V_1^2\Lambda_{2,2}\right]\Lambda_{1,1}^{-1}.$$

Then we get (2.30) by pre-multiplying the above equality on the left by $\mathbb{V}_{m+1}^T$ *and using the facts that* $\mathbb{V}_{m+1}^T V_1^i = (e_i \otimes I_r) = \widetilde{e}_i$ *for* $i = 1, 2$ *and* $\mathbb{V}_{m+1}^T A^{-1}V_1^1 = \overline{\mathbb{L}}_m (e_1 \otimes I_r) = \overline{\mathbb{L}}_m \widetilde{e}_1$.

To prove (2.31) and (2.32), we notice that for $k \geq 1, V_k = [V_k^1, V_k^2] \in \mathbb{R}^{n\times 2r}$ *and from Algorithm 4, we have*

$$\widehat{V}_{k+1} = \left[AV_k^1, A^{-1}V_k^2\right] - \mathbb{V}_k\mathbb{H}_k\left[\widetilde{e}_{2k-1}, \widetilde{e}_{2k}\right] \tag{2.34}$$

and

$$V_{k+1}H_{k+1,k} = \widehat{V}_{k+1}. \tag{2.35}$$

Using the relations (2.34) and (2.35), we obtain

$$\begin{aligned} A^{-1}V_k^2 = \widehat{V}_{k+1}\widetilde{e}_2 + \mathbb{V}_k\mathbb{H}_k\widetilde{e}_{2k} &= V_{k+1}H_{k+1,k}\widetilde{e}_2 + \mathbb{V}_k\mathbb{H}_k\widetilde{e}_{2k} \\ &= \mathbb{V}_{k+1}\overline{\mathbb{H}}_k\widetilde{e}_{2k} \end{aligned}$$

Now, multiplying on the left by $\mathbb{V}_{m+1}^T$*, we get*

$$\mathbb{V}_{m+1}^T A^{-1} V_k^2 = \mathbb{V}_{m+1}^T \mathbb{V}_{k+1} \overline{\mathbb{H}}_k \widetilde{e}_{2k}$$

hence,

$$\mathbb{V}_{m+1}^T A^{-1} \mathbb{V}_m \widetilde{e}_{2k} = \begin{bmatrix} I_{2(k+1)r} \\ 0_{2(m-k)r \times 2(k+1)r} \end{bmatrix} \overline{\mathbb{H}}_k \widetilde{e}_{2k}$$

and so

$$\overline{\mathbb{L}}_m \widetilde{e}_{2k} = \begin{bmatrix} \overline{\mathbb{H}}_k \\ 0_{2(m-k)r \times 2kr} \end{bmatrix} = \overline{\mathbb{H}}_m \widetilde{e}_{2k}$$

which gives the relation (2.31).

Now, for the even blocks, we multiply (2.34) on the left by A^{-1} *and we consider only the first* r*-columns of each block. We obtain the following relation*

$$A^{-1} \widehat{V}_{k+1}^1 = V_k^1 - A^{-1} \mathbb{V}_k \mathbb{H}_k \widetilde{e}_{2k-1}$$

Notice that since $\widehat{V}_{k+1} = V_{k+1} H_{k+1,k}$*, we also have*

$$\widehat{V}_{k+1}^1 = V_{k+1}^1 H_{k+1,k}^{1,1}$$

where $H_{k+1,k}^{1,1}$ *is the first* $r \times r$ *block of the upper* $2r \times 2r$ *triangular matrix* $H_{k+1,k}$*. Then if* $H_{k+1,k}^{1,1}$ *is nonsingular, we obtain*

$$A^{-1} V_{k+1}^1 = A^{-1} \widehat{V}_{k+1}^1 \left(H_{k+1,k}^{1,1} \right)^{-1} = \left(V_k^1 - A^{-1} \mathbb{V}_k \mathbb{H}_k \widetilde{e}_{2k-1} \right) \left(H_{k+1,k}^{1,1} \right)^{-1}$$

Multiplying from the left by $\mathbb{V}_{m+1}^T$*, we get*

$$\mathbb{V}_{m+1}^T A^{-1} V_{k+1}^1 = \left(\mathbb{V}_{m+1}^T V_k^1 - \mathbb{V}_{m+1}^T A^{-1} \mathbb{V}_k \mathbb{H}_k \widetilde{e}_{2k-1} \right) \left(H_{k+1,k}^{1,1} \right)^{-1},$$

and then

$$\begin{aligned}\mathbb{L}_{m+1}\widetilde{e}_{2k+1} &= \left(\mathbb{V}_{m+1}^T\mathbb{V}_{m+1}\widetilde{e}_{2k-1} - \mathbb{V}_{m+1}^T A^{-1}\mathbb{V}_m \begin{bmatrix} I_{2kr} \\ 0_{2(m-k)r\times 2kr}\end{bmatrix}\mathbb{H}_k\widetilde{e}_{2k-1}\right)\\ &\times \left(H_{k+1,k}^{1,1}\right)^{-1}\\ &= \left(\widetilde{e}_{2k-1} - \overline{\mathbb{L}}_m \begin{bmatrix} I_{2kr} \\ 0_{2(m-k)r\times 2kr}\end{bmatrix}\mathbb{H}_k\widetilde{e}_{2k-1}\right)\left(H_{k+1,k}^{1,1}\right)^{-1}\\ &= \left(\widetilde{e}_{2k-1} - \begin{bmatrix} \overline{\mathbb{L}}_k \\ 0_{2(m-k)r\times 2kr}\end{bmatrix}\mathbb{H}_k\widetilde{e}_{2k-1}\right)\left(H_{k+1,k}^{1,1}\right)^{-1},\end{aligned}$$

which gives the second relation (2.32).

Proposition 2.4.6. *Let* $\mathbb{V}_m = [V_1, \ldots, V_m]$, $\mathbb{L}_m := \mathbb{V}_m^T A^{-1}\mathbb{V}_m$ *be respectively the matrix and the upper block Hessenberg matrix defined in the extended block Arnoldi process and let* $\mathbb{E}_1 = [I_{2r}, 0_{2r}, \ldots, 0_{2r}]^T$. *Then for* $j = 0\ldots, m-1$, *we have the following relation*

$$A^{-j}\mathbb{V}_m\mathbb{E}_1 = \mathbb{V}_m\mathbb{L}_m^j\mathbb{E}_1 \tag{2.36}$$

and

$$\mathbb{T}_m^{-1}\mathbb{E}_j = \mathbb{L}_m\mathbb{E}_j, j = 1, \ldots, m-1 \tag{2.37}$$

Proof 2.4.6. *The relation (2.36) can be derived directly by multiplying relation (2.29) from the left by* A^{-j+1} *and from the right by* $\mathbb{E}_1$. *Then we obtain*

$$A^{-j}\mathbb{V}_m\mathbb{E}_1 = \mathbb{V}_m\mathbb{L}_m^j\mathbb{E}_1 + \sum_{i=1}^{j} A^{-(i-1)}V_{m+1}L_{m+1,m}\mathbb{E}_m^T\mathbb{L}_m^{j-i}\mathbb{E}_1$$

Now, as $\mathbb{L}_m$ *is an upper block Hessenberg matrix, it follows that* $\mathbb{E}_m^T\mathbb{L}_m^{j-i}\mathbb{E}_1 = 0$, *for* $j = 1, \ldots, m-1$.

To prove the relation (2.37), we multiply (2.29) from the right by $\mathbb{E}_j$, *and then we get*

$$A^{-1}\mathbb{V}_m\mathbb{E}_j = \mathbb{V}_m\mathbb{L}_m\mathbb{E}_j, \quad \textit{for} \quad j = 1, \ldots, m-1$$

Since, $\mathbb{V}_m$ is orthogonal, pre-multiplying the above equality by $\mathbb{V}_m^T A$, we get $\mathbb{E}_j = \mathbb{T}_m \mathbb{L}_m \mathbb{E}_j$ for $j = 1, \ldots, m-1$ and finally we obtain (2.37), if we assume that $\mathbb{T}_m$ is nonsingular.

2.4.2 Global case

As in the block case, the classical matrix (global) extended Krylov subspace is enriched by A^{-1} to get $\mathcal{K}_m^e(A, V) := \text{span}\left([V, A^{-1}V, AV\ ,\ A^{-2}V, \ldots, A^{m-1}V, A^{-m}V]\right)$. So the extended matrix Krylov subspace $\mathcal{K}_m^e(A, V)$ can be seen as a combination of two classical matrix Krylov subspaces :

$$\mathcal{K}_m^e(A, V) = \mathcal{K}_m(A, V) + \mathcal{K}_m\left(A^{-1}, A^{-1}V\right) = \mathcal{K}_m(A, V) + \mathcal{K}_{m+1}\left(A^{-1}, V\right)$$

Now using [13] let us describe the extended global Arnoldi process which builds $\{v_1, v_2, \ldots, v_{2m}\}$ $(v_i \in \mathbb{R}^{n\times r})$ an F-orthonormal basis of $\mathcal{K}_m^e(A, V)$. The two first blocks v_1 and v_2 are computed via the formulas :

$$v_1 = \frac{V}{\omega_{1,1}} \text{ and } \omega_{2,2}v_2 = \underbrace{A^{-1}V - \omega_{1,2}v_1}_{=U}, \tag{2.38}$$

where the parameters $\omega_{1,1}, \omega_{1,2}$ and $\omega_{2,2}$ are such that $n \times 2r$ matrix $[v_1, v_2]$ is F-orthonormal. Hence $\omega_{1,1} = \|V\|_F$, $\omega_{1,2} = \text{tr}\left(v_1^T\left(A^{-1}V\right)\right)$ and $\omega_{2,2} = \|U\|_F$. To compute the blocks v_{2j+1} and v_{2j+2}, for $j = 1, \ldots, m-1$, we use the following formulas

$$\begin{cases} h_{2j+1,2j-1}v_{2j+1} = Av_{2j-1} - \sum_{i=1}^{2j} h_{i,2j-1}v_i, \\ h_{2j+2,2j}v_{2j+2} = A^{-1}v_{2j} - \sum_{i=1}^{2j+1} h_{i,2j}v_i. \end{cases} \tag{2.39}$$

Using the orthogonality conditions $v_{2j+1} \perp_F v_1, \ldots, v_{2j}$ and $v_{2j+2} \perp_F v_1, \ldots, v_{2j+1}$, we obtain

$$h_{i,2j-1} = \text{tr}\left(v_i^T A v_{2j-1}\right) = \text{tr}\left(v_i^T U_{i,1}^j\right) \quad \text{and} \quad h_{i,2j} = \text{tr}\left(v_i^T A^{-1} v_{2j}\right) = \text{tr}\left(v_i^T U_{i,2}^j\right),$$

where $U_{i,1}^j = Av_{2j-1} - \sum_{k=1}^{i-1} h_{k,2j-1}v_k$ and $U_{i,2}^j = A^{-1}v_{2j} - \sum_{k=1}^{i-1} h_{k,2j}v_k$. The parameters $h_{2j+1,2j-1}$ and $h_{2j+2,2j}$ are such that $\|v_{2j+1}\|_F = 1$ and $\|v_{2j+2}\|_F = 1$

respectively. Hence,

$$h_{2j+1,2j-1} = \left\|U^j_{2j+1,1}\right\|_F \quad \text{and} \quad h_{2j+2,2j} = \left\|U^j_{2j+1,2}\right\|_F .$$

Now, to describe a new way for computing the column vectors of the $n \times 2mr$ matrix $\mathcal{V}_m := [v_i]_{i=1,\ldots,2m}$ and the nonzeros entries of the upper block Hessenberg matrix $\mathcal{H}_m := [h_{i,j}]^{j=1,\ldots,2m}_{i=1,\ldots,2j+2}$, we introduce

$$V_i = [v_{2i-1}, v_{2i}] = \left[V_i^1, V_i^2\right] \text{ and } H_{i,j} = \begin{pmatrix} h_{2i-1,2j-1} & h_{2i-1,2j} \\ h_{2i,2j-1} & h_{2i,2j} \end{pmatrix}, \tag{2.40}$$

which are respectively the i th $n \times 2r$ block vector of the matrix $\mathcal{V}_m$ and the (i,j) th 2×2 block of the upper block Hessenberg matrix $\mathcal{H}_m$. Using the Kronecker product and the formulas in (2.39), (2.40) we obtain

$$V_{j+1}\left(H_{j+1,j} \otimes I_r\right) = \left[AV_j^1, A^{-1}V_j^2\right] - \sum_{i=1}^{j} V_i \left(H_{i,j} \otimes I_r\right)$$

As the $n \times 2r$ blocks $V_1, \ldots, V_m$ are orthogonal with respect to the $\diamond$-product (i.e., $V_i^T \diamond V_j = O_2$ for $i \neq j$), then using the properties of the $\diamond$-product, we see that

$$H_{i,j} = V_i^T \diamond \left[AV_j^1, A^{-1}V_j^2\right] = V_i^T \diamond U_i^j \quad \text{for } i = 1, \ldots, j,$$

where $U_i^j = \sum_{k=1}^{i-1} V_k \left(H_{k,j} \otimes I_r\right)$. For the upper triangular block $H_{j+1,j}$, we recall that $H_{j+1,j}$ and V_{j+1} are computed such that $V_{j+1} \diamond V_i = 0_2$ for $i = 1, \ldots, j$ and $V_{j+1} \diamond V_{j+1} = I_2$, so we get V_{j+1} and $H_{j+1,j}$ by computing the global QR factorization of U^j_{j+1}.

Let $\Omega = [\omega_{i,j}]$ be the 2×2 upper triangular matrix whose nonzero entries are defined in (2.38), we easily check that Ω and V_1 the first block of the $n \times mr$ matrix $\mathcal{V}_m$ can be obtained by the global QR decomposition of $[V, A^{-1}V]$.

Finally, summarizing the previous results, we describe the extended global Arnoldi process as follows :

Provided that the upper 2×2 triangular matrices $H_{j+1,j} (j = 1, \ldots, m)$ are full rank, the above process computes an $n \times 2mr$ F-orthonormal matrix $\mathcal{V}_m =$

Algorithm 5 The extended global Arnoldi algorithm

Require: $A \in \mathbb{R}^{n\times n}$, $V \in \mathbb{R}^{n\times r}$, m.
  Compute $[V, A^{-1}V] = V_1(\Omega \otimes I_r)$; (global QR decomposition) $\mathcal{V}_1 = [V_1]$.
  for $j = 1, \ldots, m$ **do**
    Set $V_j^{(1)}$: first r columns of V_j ; $V_j^{(2)}$: second r columns of V_j.
    $\widetilde{V}_{j+1} = [AV_j^{(1)}, A^{-1}V_j^{(2)}]$.
    F-Orthogonalize $\widetilde{V}_{j+1}$ with respect to $\mathcal{V}_1, \ldots, \mathcal{V}_j$ to get V_{j+1}, i.e.,
    for $i = 1, 2, \ldots, j$ **do**
      $H_{i,j} = (V_i)^T \diamond V_{j+1}$.
      $\widetilde{V}_{j+1} = \widetilde{V}_{j+1} - V_i(H_{i,j} \otimes I_r)$.
    end for
    Compute $\widetilde{V}_{j+1} = V_{j+1}(H_{j+1,j} \otimes I_r)$; (global QR decomposition).
    $\mathcal{V}_{j+1} = [\mathcal{V}_j, V_{j+1}]$.
  end for

$[V_1, \ldots, V_m]$ with $V_i \in \mathbb{R}^{n\times 2r} (i = 1, \ldots, m)$ and a $2(m+1) \times 2m$ upper block Hessenberg matrix $\overline{\mathcal{H}}_m = [H_{i,j}]$ with $H_{i,j} \in \mathbb{R}^{2\times 2}$.

We now introduce the $2m \times 2m$ matrix given by $\mathcal{T}_m = \mathcal{V}_m^T \diamond (A\mathcal{V}_m) = [T_{i,j}]$, where $T_{i,j} = V_i^T \diamond (AV_j) \in \mathbb{R}^{2\times 2}$, for $i, j = 1, \ldots, m$. Note that as $\{v_1, \ldots, v_{2m}\}$ is an F-orthonormal basis and $A\mathcal{K}_j \subset \mathcal{K}_{j+1}$ then $\operatorname{tr}\left(v_i^T A v_{2j+1}\right) = 0$ (for $j = 0, \ldots, m-1$, $i > 2j+3$) and $\operatorname{tr}\left(v_i^T A v_{2j}\right) = 0$ (for $j = 1, \ldots, m, i > 2j+1$). Hence $\mathcal{T}_m$ is a $2m \times 2m$ block upper Hessenberg matrix with 2×2 blocks. In addition, the lower diagonal blocks $T_{j+1,j}$ have zero second row.

Let $\overline{\mathcal{T}}_m = \mathcal{V}_{m+1}^T \diamond (A\mathcal{V}_m)$ and $E_m^T = \left[O_{2\times 2(m-1)}, I_2\right]$ be the matrix of the last 2 rows of the $2m \times 2m$ identity matrix I_{2m}, we notice that using similar arguments as those used in the proof of Proposition (2.4.3) , we can show that after m steps, of Algorithm 5, we have

$$A\mathcal{V}_m = \mathcal{V}_{m+1}\left(\overline{\mathcal{T}}_m \otimes I_r\right) = \mathcal{V}_m\left(\mathcal{T}_m \otimes I_r\right) + V_{m+1}\left(T_{m+1,m}E_m^T \otimes I_r\right)$$

Next, the following result gives a recursion for computing $\overline{\mathcal{T}}_m$ without requiring additional matrix-vector products with A.

Proposition 2.4.7. *Let $\overline{\mathcal{T}}_m = [t_{:,1}, \ldots, t_{:2m}]$ and $\overline{\mathcal{H}}_m = [h_{:,1}, \ldots, h_{:2m}]$ where $t_{:,i}, h_{:,i} \in \mathbb{R}^{2(m+1)}$ are the i th column of the $2(m+1)\times 2m$ upper block Hessenberg matrices $\overline{\mathcal{T}}_m$ and $\overline{\mathcal{H}}_m$ respectively. Then the odd columns are such that*

$$t_{:,2j-1} = h_{:,2j-1}, \quad \textit{for } j = 1, \ldots, m$$

while the even columns satisfy

$$t_{:,2} = \frac{1}{\omega_{2,2}} \left(\omega_{1,1} e_1^{2(m+1)} - \omega_{1,2} h_{:,1} \right),$$
$$t_{:,2j+2} = \frac{1}{h_{2j+2,2j}} \left(e_{2j}^{2(m+1)} - t_{:,1:2j+1} h_{1:2j+1,2j} \right) \quad \textit{for } j = 1, \ldots, m-1,$$

where $e_i^{(k)}$ *is the ith column vector of the identity matrix* I_k *and* $w_{1,1}, w_{1,2}$ *are defined in (2.38).*

Proof 2.4.7. *From the first relation in (2.39), we have* $Av_{2j-1} = \sum_{i=1}^{2j+1} h_{i,2j-1} v_i = \mathcal{V}_{m+1} \left(h_{:,2j-1} \otimes I_r \right)$*. Hence, using the properties of the* $\diamond$*-product we obtain*

$$t_{:,2j-1} = \mathcal{V}_{m+1} \diamond (Av_{2j-1}) = h_{:,2j-1}, \quad \textit{for } j = 1, \ldots, m$$

Using (2.38), we obtain $\omega_{2,2} Av_2 = \omega_{1,1} v_1 - \omega_{1,2} Av_1 = \omega_{1,1} v_1 - \omega_{1,2} \mathcal{V}_{m+1} \left(h_{:,1} \otimes I_r \right)$*. Then, premultiplying on the left the last equality by* $\mathcal{V}_{m+1}^T$*, we get*

$$t_{:,2} = \mathcal{V}_{m+1}^T \diamond (Av_2) = \frac{1}{\omega_{2,2}} \left(\omega_{1,1} e_1^{2(m+1)} - \omega_{1,2} h_{:,1} \right)$$

Now, premultiplying by A*, the second formula in (2.39) gives for* $j = 1, \ldots, m$*,*

$$\begin{aligned} h_{2j+2,2j} Av_{2j+2} &= v_{2j} - \sum_{i=1}^{2j+1} h_{i,2j} Av_i \\ &= v_{2j} - A \left[v_1, \ldots, v_{2j+1} \right] \left(h_{1:2j+1,2j} \otimes I_r \right). \end{aligned}$$

Since $t_{:,2j+2} = \mathcal{V}_{m+1}^T \diamond (Av_{2j+2})$*, we get*

$$t_{:,2j+2} = \frac{1}{h_{2j+2,2j}} (e_{2j}^{2(m+1)} - \underbrace{\mathcal{V}_{m+1} \diamond \left(A \left[v_1, \ldots, v_{2j+1} \right] \right)}_{= t_{:,1:2j+1}} h_{1:2j+1,2j}),$$

which completes the proof.

Now, as for the block case seen in the previous subsection, we consider the matrix
$\mathcal{L}_m = \left[L_{i,j}^g \right] = [l_{p,q}]$ defined by

$$\mathcal{L}_m = \mathcal{V}_m^T \diamond \left(A^{-1} \mathcal{V}_m \right)$$

where $L_{i,j}^g \in \mathbb{R}^{2\times 2}$ for $i, j = 1, \ldots, m$ and $l_{p,q} \in \mathbb{R}$ for $p, q = 1, \ldots, 2m$. We

mention that it can be easily verified that the matrix $\mathcal{L}_m$ is a $2m \times 2m$ upper block Hessenberg matrix.

Proposition 2.4.8. *Assume that m steps of Algorithm 5 have been run and let $\overline{\mathcal{L}}_m = \mathcal{V}_{m+1}^T A^{-1} \mathcal{V}_m$, then we have the following relations*

$$A^{-1}\mathcal{V}_m = \mathcal{V}_{m+1}\left(\overline{\mathcal{L}}_m \otimes I_r\right) \tag{2.41}$$

$$= \mathcal{V}_m\left(\mathcal{L}_m \otimes I_r\right) + V_{m+1}^g\left(L_{m+1,m}E_m^T \otimes I_r\right) \tag{2.42}$$

The sub-matrices $L_{i+1,i}^g \in \mathbb{R}^{2\times 2}$ are such that the first column is zero. So, the sub-matrix $L_{m+1,m}^g$ is such that $l_{2m+1,2m-2} = l_{2m+2,2m-2} = 0$, i.e.,

$$L_{m+1,m}^g = \begin{bmatrix} 0 & l_{2m+1,2m} \\ 0 & l_{2m+2,2m} \end{bmatrix} \tag{2.43}$$

And notice that since $V_{m+1} = \left[V_{m+1}^1, V_{m+1}^2\right]$, then by using (2.43) the Arnoldi relation (2.41) becomes

$$A^{-1}\mathcal{V}_m = \mathcal{V}_m\mathcal{L}_m + \left[O_{n\times(2m-1)}, l_{2m+1,2m}V_{m+1}^{(1)} + l_{2m+2,2m}V_{m+1}^{(2)}\right]$$

Now, in order to update progressively the columns of the matrix $\overline{\mathcal{L}}_m$ without inverting A or solving linear systems with A, we recall some elementary results :

— Let $[V, A^{-1}V] = V_1\left(\Omega \otimes I_r\right)$ be the global QR decomposition of $[V, A^{-1}V]$ which can be written as

$$\left[V, A^{-1}V\right] = V_1^g\left(\Omega \otimes I_r\right) = \left[V_1^1, V_1^2\right]\begin{bmatrix} \Omega_{1,1} & \Omega_{1,2} \\ 0 & \Omega_{2,2} \end{bmatrix}$$

— For $k = 1, \ldots, m$, let us partition the lower triangular matrix $H_{k+1,k}^g$ under the form

$$H_{k+1,k} = \begin{bmatrix} h_{2k+1,2k-1} & h_{2k+1,2k} \\ 0 & h_{2k+2,2k} \end{bmatrix}.$$

As in the block case, the following result enables us to compute $\overline{\mathbb{L}}_m$ directly from the columns of the upper block Hessenberg matrix $\overline{\mathbb{H}}_m$ obtained from

Algorithm 5.

Proposition 2.4.9. *Let* $\overline{\mathcal{L}}_m = [l_{:,1}, \ldots, l_{:,2m}]$ *and* $\overline{\mathcal{H}}_m = [h_{:,1}, \ldots, h_{:,2m}]$ *be the upper block Hessenberg matrices defined earlier. Then we have the following relations*

$$l_{:,1} = \left(\Omega_{1,2}e_1 + \Omega_{2,2}e_2\right)/\Omega_{1,1} \tag{2.44}$$

and for $k = 1, \ldots, m$*, we have*

$$\bar{l}_{:,2k} = h_{:,2k} \tag{4.11}$$

and

$$l_{2k+1} = \left(e_{2k-1} - \begin{bmatrix} \overline{\mathbb{L}}_k \\ 0_{2(m-k)\times 2k} \end{bmatrix} \mathbb{H}_k e_{2k-1}\right)/h_{2k+1,2k}, \tag{2.45}$$

where the e_i*'s are the vectors of the canonical basis.*

Proposition 2.4.10. *Let* $\mathcal{V}_m = [V_1, \ldots, V_m]$, $\mathcal{L}_m := \mathcal{V}_m^T \diamond (A^{-1}\mathcal{V}_m)$ *be respectively the matrix and the upper block Hessenberg matrix defined by the extended global Arnoldi process and let* $E_1 = [I_2, 0_2, \ldots, 0_2]^T$*. Then for* $j = 1 \ldots, m-1$*, we have the following relation*

$$A^{-j}\mathcal{V}_m\mathbb{E}_1 = A^{-j}\mathcal{V}_m\left(E_1 \otimes I_r\right) = \mathcal{V}_m\left(\mathcal{L}_m^j E_1 \otimes I_r\right) \tag{2.46}$$

and for $j = 1, \ldots, m-1$ *we have*

$$\mathcal{T}_m^{-1}E_j = \mathcal{L}_m E_j \tag{2.47}$$

Proof 2.4.8. *Pre-multiplying (2.41) by* A^{-j+1} *and using the properties of the Kronecker product we get*

$$A^{-j}\mathcal{V}_m = \mathcal{V}_m\left(\mathcal{L}_m^j \otimes I_r\right) + \sum_{i=1}^{j} A^{-(i-1)}V_{m+1}\left(L_{m+1,m}E_m^T\mathcal{L}_m^{j-i} \otimes I_r\right)$$

Post-multiplying the above equality by $(E_1 \otimes I_r)$*, we obtain*

$$
\begin{aligned}
A^{-j}\mathcal{V}_m\mathbb{E}_1 &= A^{-j}\mathcal{V}_m\left(E_1 \otimes I_r\right) \\
&= \mathcal{V}_m\left(\mathcal{L}_m^j E_1 \otimes I_r\right) + \sum_{i=1}^{j} A^{-(i-1)} V_{m+1}\left(L_{m+1,m} E_m^T \mathcal{L}_m^{j-i} E_1 \otimes I_r\right), \\
&= \mathcal{V}_m\left(\mathcal{L}_m^j E_1 \otimes I_r\right).
\end{aligned}
$$

In the last equality, we used the fact that $E_m^T \mathcal{L}_m^{j-i} E_1 = 0$ since $\mathcal{L}_m$ is an upper block Hessenberg matrix.

Next, we give a general result that is satisfied by upper Hessenberg matrices. This result will be used when establishing moment matching properties for the block and global Arnoldi processes.

Proposition 2.4.11. *[1] Let $T = (T_{i,j})$ and $L = (L_{i,j})$ be two upper block Hessenberg matrices with blocks $T_{i,j}, L_{i,j} \in \mathbb{R}^{r\times r}$ for $i, j = 1, \ldots, m$ and suppose that*

$$
T\mathbb{E}_j = L\mathbb{E}_j, \quad \textit{for} \quad j = 1, \ldots, m-1 \tag{2.48}
$$

where $\mathbb{E}_j = [0_r, \ldots, 0_r, I_r, 0_r, \ldots, 0_r]^T$, is the $mr \times r$ matrix whose columns are the column $j, \ldots, j+r$ of the identity matrix I_{mr}. Then

$$
T^k\mathbb{E}_1 = L^k\mathbb{E}_1, \textit{ for } k = 1, \ldots, m-1 \tag{2.49}
$$

2.4.3 Application to model reduction problems

We consider the following Linear Time Independent (LTI) dynamical system

$$
\begin{cases}
\dot{x}(t) = Ax(t) + Bu(t), \\
y(t) = Cx(t),
\end{cases} \tag{2.50}
$$

where $x(t) \in \mathbb{R}^n$ is the state vector, $u(t), y(t) \in \mathbb{R}^r$ are the input and the output vectors of the system (), respectively. The matrices B, C^T are in $\mathbb{R}^{n\times r}$ and $A \in \mathbb{R}^{n\times n}$ is assumed to be large and sparse. The transfer function of the original system (2.50) is given as

$$F(s) = C\left(sI_n - A\right)^{-1} B.$$

In many applications, the dimension n of the system (2.22) is large which makes the computations infeasible in terms of execution time and memory. Then the goal of model reduction problems is to produce a low-order system of the form

$$\begin{cases} \dot{x}_m(t) = A_m x_m(t) + B_m u(t) \\ y_m(t) = C_m x_m(t) \end{cases} \tag{2.51}$$

where $A_m \in \mathbb{R}^{p\times p}, B_m, C_m^T \in \mathbb{R}^{p\times r}$. The basic technique is to project the system's state space of dimension n onto a space of lower dimension $p \ll n$, in such a way that the reduced-order model preserves the important properties of the original system like stability and passivity and such that the output y_m is close to the output y of the original system. The associated low-order transfer function is denoted by

$$F_m(s) = C_m\left(sI_p - A_m\right)^{-1} B_m.$$

The transfer function F relates the Laplace transform of the output vector to that of the input vector. For that reason, it is called the transfer function matrix of the system. Each entry $F_{i,j}(s)$ is a rational function representing the transfer function between the i-th input and the j-th output, all other inputs being set equal to zero.

The rational function F can be expressed as a sum of a Taylor series around $(s = \infty)$ in the following form

$$F(s) = \frac{1}{s} C\left(I_n - \frac{A}{s}\right)^{-1} B = \frac{1}{s}\sum_{i=0}^{\infty} M_i s^{-i}, \text{ with } M_i = CA^iB.$$

Recall that the matrix coefficients M_i are called the Markov parameters of F.

Now applying the extended block Arnoldi process to the pair (A, B), we can verify that the original transfer function F can be approximated by

$$\mathbb{F}_m(s) = \mathbb{C}_m\left(sI_{2mr} - \mathbb{T}_m\right)^{-1} \mathbb{B}_m$$

where $\mathbb{T}_m = \mathbb{V}_m^T A \mathbb{V}_m, \mathbb{C}_m = C\mathbb{V}_m$ and $\mathbb{B}_m = \mathbb{V}_m^T B$. This reduced transfer function is related to the low-order dynamical system (2.51) with $A_m = \mathbb{T}_m$.

Similarly, if m iterations of the extended global Arnoldi algorithm are applied to the pair (A, B), then we can approximate F by

$$\mathcal{F}_m(s) = \mathcal{C}_m \left(sI_{2mr} - (\mathcal{T}_m \otimes I_r)\right)^{-1} \mathcal{B}_m$$

where $\mathcal{T}_m = \mathcal{V}_m^T \diamond (A\mathcal{V}_m), \mathcal{C}_m = C\mathcal{V}_m$ and $\mathcal{B}_m = \mathcal{V}_m^T \diamond B = \|B\|_F \left(e_1^{(2m)} \otimes I_r\right)$. In this case, the reduced transfer function is related to the low-order dynamical system (2.51) with $A_m = \mathcal{T}_m \otimes I_r$. The developments of $\mathbb{F}_m$ and $\mathcal{F}_m$ around $s = \infty$ give the following expressions

$$\mathbb{F}_m(s) = \frac{1}{s} \sum_{i=0}^{\infty} m_i^b s^{-i}, \text{ with } m_i^b = \mathbb{C}_m \mathbb{T}_m^i \mathbb{B}_m$$

and

$$\mathcal{F}_m(s) = \frac{1}{s} \sum_{i=0}^{\infty} m_i^g s^{-i}, \text{ with } m_i^g = \mathcal{C}_m \left(\mathcal{T}_m \otimes I_r\right)^i \mathcal{B}_m$$

In this case, one can show that the first m Markov parameters are matched, i.e. in the block case

$$M_i = m_i^b, \quad i = 0, \ldots, m-1$$

and in the global case

$$M_i = m_i^g, \quad i = 0, \ldots, m-1.$$

Now, the development of the Neumann series of F around $s = 0$ gives the following expression

$$F(s) = \sum_{i=0}^{\infty} \widetilde{M}_{i+1} s^i$$

The matrix coefficients $\widetilde{M}_i$ are called the moments of F and they are given by

$$\widetilde{M}_j = -CA^{-j}B, j = 1, 2, \ldots$$

By considering the Taylor series of $\mathbb{F}_m$ and $\mathcal{F}_m$, we get the following expansion of $\mathbb{F}_m$ around $s = 0$

$$\mathbb{F}_m(s) = \sum_{i=0}^{\infty} \widetilde{m}_{i+1}^b s^i, \text{ with } \widetilde{m}_i^b = -\mathbb{C}_m \mathbb{T}_m^{-i} \mathbb{B}_m$$

while for $\mathcal{F}_m$, we get

$$\mathcal{F}_m(s) = \sum_{i=0}^{\infty} \widetilde{m}_{i+1}^g s^i, \text{ with } \widetilde{m}_i^b = -\mathcal{C}_m \left(\mathcal{T}_m \otimes I_r\right)^{-i} \mathcal{B}_m$$

As for the Markov parameters, the following result shows that the first m moments resulting from the Newman series of the transfer function F around $s = 0$ are also matched either by those of $\mathbb{F}_m$ when using the extended block Arnoldi process or by those of $\mathcal{F}_m$ when using the extended global Arnoldi process.

Proposition 2.4.12. *Let $\widetilde{M}_j$ and $\widetilde{m}_j^b$ be the matrix moments given by the Newman expansions of F and $\mathbb{F}_m$, respectively around $s = 0$. Then we have*

$$\widetilde{M}_j = \widetilde{m}_j^b, \quad \textit{for } j = 0, \ldots, m-1$$

Proof 2.4.9. *The equality is verified for $j = 0$. For $j \geq 1$, we obtain*

$$\begin{aligned} \widetilde{M}_j &= CA^{-j}B \\ &= CA^{-j}V_1 \begin{bmatrix} \Lambda_{1,1} \\ 0 \end{bmatrix} = CA^{-j}\mathbb{V}_m \mathbb{E}_1 \begin{bmatrix} \Lambda_{1,1} \\ 0 \end{bmatrix} \end{aligned}$$

Therefore, using the result of Proposition 2.4.6, we get

$$\widetilde{M}_j = C\mathbb{V}_m \mathbb{L}_m^j \mathbb{E}_1 \begin{bmatrix} \Lambda_{1,1} \\ 0 \end{bmatrix}$$

On the other hand, applying Proposition () to the upper Hessenberg matrices $\mathbb{L}_m$ and $\mathbb{T}_m^{-1}$, we get

$$\mathbb{L}_m^j \mathbb{E}_1 = \mathbb{T}_m^{-j} \mathbb{E}_1; \quad j = 0, \ldots, m-1$$

and this gives for $j = 1, \ldots, m-1$

$$\begin{aligned} \widetilde{M}_j &= C\mathbb{V}_m \mathbb{T}_m^{-j} \mathbb{V}_m^T V_1 \begin{bmatrix} \Lambda_{1,1} \\ 0 \end{bmatrix} \\ &= C\mathbb{V}_m \mathbb{T}_m^{-j} \mathbb{V}_m^T B = \mathbb{C}_m \mathbb{T}_m^{-j} \mathbb{B}_m \\ &= \widetilde{m}_j^b \end{aligned}$$

Now, using the extended global Arnoldi process, we can also state the following result

Proposition 2.4.13. *Let* $\widetilde{M}_j$ *and* $\widetilde{m}_j^g$ *be the matrix moments given by the Newman expansions of* F *and* $\mathcal{F}_m$*, respectively around* $s = 0$*. Then we have*

$$\widetilde{M}_j = \widetilde{m}_j^g, \quad \textit{for } j = 0, \ldots, m-1$$

Proof 2.4.10. *The equality is verified for* $j = 0$*. For* $j \geq 1$*, we obtain*

$$\begin{aligned} \widetilde{M}_j &= CA^{-j}B \\ &= CA^{-j}V_1 \begin{bmatrix} \Omega_{1,1} \\ 0 \end{bmatrix} = CA^{-j}\mathcal{V}_m \mathbb{E}_1 \begin{bmatrix} \Omega_{1,1} \\ 0 \end{bmatrix} \end{aligned}$$

Therefore, using the result of Proposition (...), we get

$$\widetilde{M}_j = C\mathcal{V}_m \left(\mathcal{L}_m^j E_1 \otimes I_r \right) \begin{bmatrix} \gamma_{1,1} \\ 0 \end{bmatrix}$$

Now, similarly to the block case, applying proposition 2.4.10 to $\mathcal{L}_m$ *and* $\mathcal{T}_m^{-1}$*,*

we also have $\mathcal{L}_m^j E_1 = \mathcal{T}_m^{-j} E_1$ *for* $j = 0, \ldots, m-1$ *and so we get*

$$\begin{aligned}
\widetilde{M_j} = C\mathcal{V}_m \left(\mathcal{T}_m^{-j} E_1 \otimes I_r\right) \begin{bmatrix} \Omega_{1,1} \\ 0 \end{bmatrix} &= C\mathcal{V}_m \left(\mathcal{T}_m^{-j} \otimes I_r\right) (E_1 \otimes I_r) \begin{bmatrix} \Omega_{1,1} \\ 0 \end{bmatrix} \\
&= C\mathcal{V}_m \left(\mathcal{T}_m^{-j} \otimes I_r\right) \left(\mathcal{V}_m^T \diamond V_1\right) \begin{bmatrix} \Omega_{1,1} \\ 0 \end{bmatrix} \\
&= C\mathcal{V}_m \left(\mathcal{T}_m^{-j} \otimes I_r\right) \left(\mathcal{V}_m^T \diamond \left(V_1 \begin{bmatrix} \Omega_{1,1} \\ 0 \end{bmatrix} \otimes I_r\right)\right) \\
&= C\mathcal{V}_m \left(\mathcal{T}_m^{-j} \otimes I_r\right) \left(\mathcal{V}_m^T \diamond B\right) \\
&= \mathcal{C}_m \mathcal{T}_m^{-j} \mathcal{B}_m = \widetilde{m}_j^g.
\end{aligned}$$

2.5 Numerical tests

In this section, we present experimental results to demonstrate the effectiveness of our methods, and provide a comparison between classical Krylov subspace methods (Global and Block) and extended Krylov methods (Global and Block).
All the experiments were performed on a computer of Intel Core i7 at 2.3GHz and 16GB of RAM. The algorithms were coded using Python.

Exemple 1. We consider the matrices A obtained from the five-point central finite difference discretization of the two-dimensional variable-coefficient linear elliptic equation (pde model)

$$L_A(u) = -\left(e^{xy} u_x\right)_x - \left(e^{xy} u_y\right)_y + 20(x+y)u_x + ((x+y)u_y) + \frac{1}{1+x+y} u.$$

The domain is the unit square $[0,1] \times [0,1]$ with Dirichlet boundary conditions. The number of grid points along x axis and y axis is $N_x = 47$ and $N_y = 63$ respectively, the dimension of the matrices A is $n = N_x \times N_y = 2961$.

In this experiment, we consider the FDM matrix A as given in the example 1 to test classical Arnoldi method in the block and global case. The matrices B and C of sizes $n \times p$ and $p \times n$, respectively, where random matrices with entries uniformly distributed in $[0,1]$.

Using classical global Arnoldi method, we compare the H_∞ norm of the fre-

quency response for the original system and its approximation with m=10 p=6 for the frequencies $\omega \in [10^{-5}, 10^{5}]$, we get the plot on the left of figure 2.1, and on the right we plot the H_∞ norms of the errors $\|F(i\omega) - F_m(i\omega)\|_{H_\infty}$.

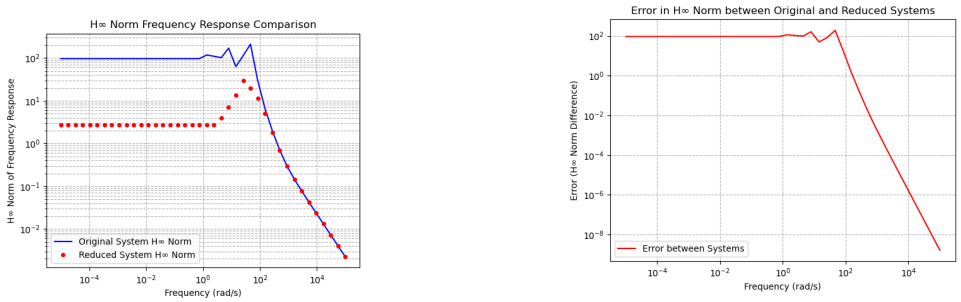

FIGURE 2.1 – Classical global Arnoldi method. Left : Comparison of the H_∞ norm of Frequency Response of the original and approximated system. Right : The H_∞ Error norms $\|F(i\omega) - F_m(i\omega)\|_{H_\infty}$.

Now, we will use the block case of the classical Arnoldi method, to get approximated system of the original system, and in the left of figure 2.2 we compare the H_∞ norms of the frequency response of the original system and its approximation, and in the right of the figure we plot the H_∞ error norm $\|F(i\omega) - F_m(i\omega)\|_{H_\infty}$.

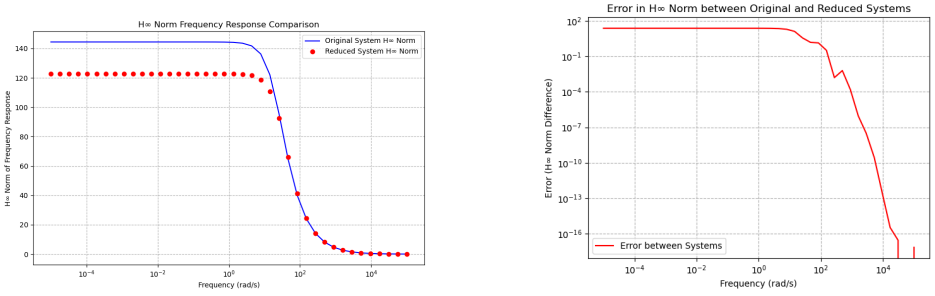

FIGURE 2.2 – Classical block Arnoldi method. Left : comparison of the H_∞ norm of frequency response of the original and approximated system. Right : The H_∞ error norms $\|F(i\omega) - F_m(i\omega)\|_{H_\infty}$.

Example 2. We considered FDM model, the corresponding matrix A is obtained from the finite difference discretization of the operator

$$L_A(u) = \Delta u - f(x,y)\frac{\partial u}{\partial x} - g(x,y)\frac{\partial u}{\partial y} - h(x,y)u,$$

on the unit square $[0,1] \times [0,1]$ with homogeneous Dirichlet boundary conditions with

$$f(x,y) = \sin(x+2y),$$
$$g(x,y) = e^{x+y},$$
$$h(x,y) = x+y,$$

and the matrices B and C of sizes $n \times p$ and $p \times n$, respectively, where random matrices with entries uniformly distributed in $[0,1]$. The number of inner grid points in each direction was $n_0 = 100$ and the dimension of A is $n = n_0^2$.

The left plots of Figure 2.1, shows a comparison of the H_∞ norm of Frequency Response of the original and approximated system using extended global Arnoldi method with $m = 30$ and $p = 8$ for the frequencies $\omega \in [10^5, 10^5]$.
We also plot in the right of Figure 2.2, the maximum singular value of the error between the frequency response of the original and approximated system which is the H_∞ error-norms $\|F(i\omega) - F_m(i\omega)\|_{H_\infty}$ corresponding to the extended Global Arnoldi methods with $m = 30$ and $p = 8$ for the frequencies $\omega \in [10^{-5}, 10^5]$.

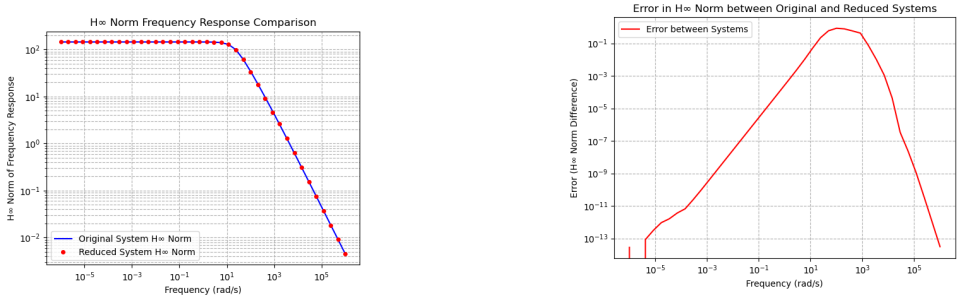

FIGURE 2.3 – Extended Global Arnoldi Method. Left : Comparison of the H_∞ norm of Frequency Response of the original and approximated system. Right : The H_∞ Error norms $\|F(i\omega) - F_m(i\omega)\|_{H_\infty}$.

The left plots of Figure 2.2, shows a comparison of the H_∞ norm of frequency response of the original and approximated system using extended block Arnoldi method with $m = 20$ and $p = 6$ for the frequencies $\omega \in [10^5, 10^5]$.
In the right of Figure 2.2 we plot the maximum singular value of the error between the frequency response of the original and approximated system which is the H_∞ error-norms $\|F(i\omega) - F_m(i\omega)\|_{H_\infty}$ corresponding to the extended block Arnoldi methods with $m = 30$ and $p = 6$ for the frequencies

$\omega \in [10^{-5}, 10^{5}]$.

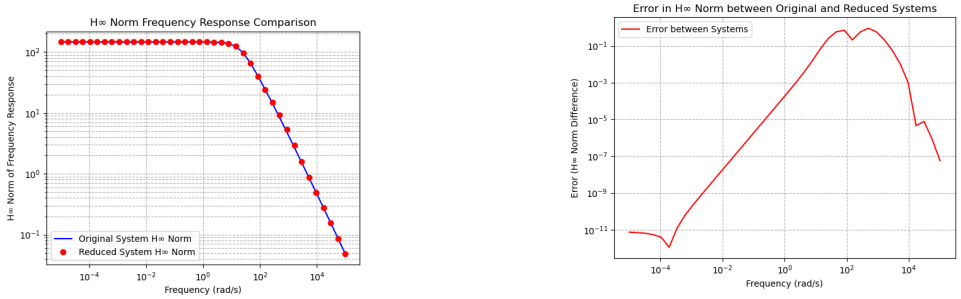

FIGURE 2.4 – Extended Block Arnoldi Method. Left : Comparison of the H_∞ norm of Frequency Response of the original and approximated system. Right : The H_∞ Error norms $\|F(i\omega) - F_m(i\omega)\|_{H_\infty}$.

In the last figure we give a comparison of the H_∞ Error norms $\|F(i\omega) - F_m(i\omega)\|_{H_\infty}$, with $\omega \in [10^{-5}, 10^{5}]$ using the four methods :

EBA : Extended block Arnoldi method, m=10, p=8

EGA : Extended global Arnoldi method m=30, p=8

CGA : Classical global Arnoldi method m=5, p=8

CBA : Classical block Arnoldi method m=45, p=8

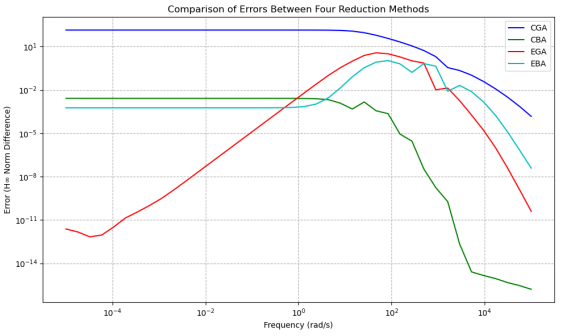

FIGURE 2.5 – Comparison of the H_∞ error norms $\|F(i\omega) - F_m(i\omega)\|_{H_\infty}$ using the four methods

Model / Method	CBA	CGA	EBA	EGA
FDM1, $n = 10000, p = 8$				
H_∞-Error norms	2.78	763.3	3.11	3.65
Space dimension	360	160	160	60
FDM2, $n = 2961, p = 6$				
H_∞-Error norms	0.003	140.74	0.7	0.87
Space dimension	270	120	120	60

TABLE 2.1 – The H_∞ error-norms $\|F - F_m\|_{H_\infty}$, execution times and reduced space dimensions for different methods with the frequencies $\omega \in [10^{-5}, 10^{-2}]$

As we can see, the classical Krylov subspace based methods have limitations compared to the extended Krylov subspace based methods for model order reduction in MIMO systems.

Now, let's explore the efficiency of the extended Krylov subspace methods in the following example :

Exemple 3. We considered the models `CDplayer` and `FOM` benchmark models listed in Table 2.2, and were obtained from NICONET [17]. Although the matrices of these models have small sizes, they are usually considered as benchmark examples.

TABLE 2.2 – Test matrices

Matrix A	**size** n	$\|A\|_F$	cond(A)
FOM	1006	$1.82e+04$	1000
CDplayer	120	$2.31e+05$	$1.81e+04$

The plots of Figure 3.11 show the norms of the errors $\|F(i\omega) - F_m(i\omega)\|_2$ for the extended block (dashed), extended global (solid), and the balanced-truncation (dashed-dotted) methods which is one of the most known methods in model reduction with $\omega \in [10^{-5}, 10^5]$.

Figure 3.11 shows that the three methods return similar results with an advantage, in the right plots of this figure, for balanced truncation for medium frequencies. However, balanced truncation is generally more expensive as compared to the two other methods.

Note that for balanced truncation we used "balred" from "control" library in Python.

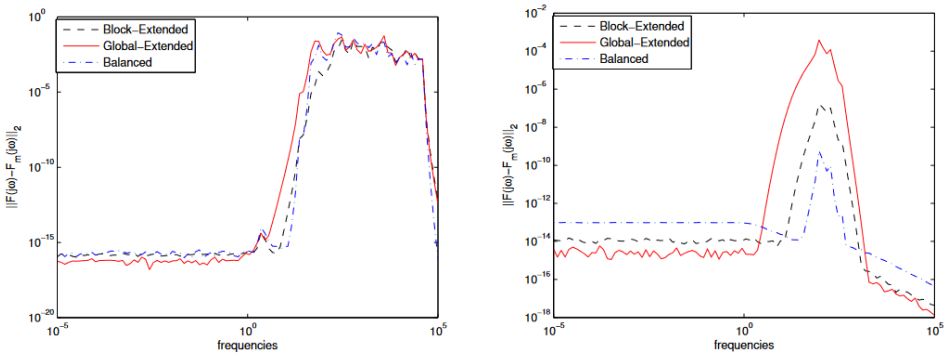

FIGURE 2.6 – The norms of the errors $\|F(i\omega) - F_m(i\omega)\|_2$ for the extended block (dashed), extended global (solid), and the balanced-truncation (dashed-dotted) methods with $\omega \in [10^{-5}, 10^5]$. Left : the `CDplayer` model with $m = 10$ and $r = 2$. Right : the `FOM` model with $m = 15$, $r = 3$.

Chapitre 3

Multi-linear dynamical systems

3.1 Introduction to tensor matrix

3.1.1 Definitions

A tensor is a multi-way array or multi-dimensional matrix. The tensor order is the number of its indices, which is called modes or ways.
The tensors are obviously generalizations of scalars (a 0th-order tensor), vectors (1st-order tensor) and matrices (2nd-order tensor).
To deepen our understanding of tensors we give this definition :

Definition 3.1.1. ***Tensors***

Let $I_1, I_2, \ldots, I_N \in \mathbb{N}$, $\mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ *is an* N^{th}*-order tensor of size* $I_1 \times I_2 \times \ldots \times I_N$. *whose elements are denoted by* $\mathcal{Y}_{i_1,i_2,\ldots,i_N}$ *or* $\mathcal{Y}(i_1, i_2, \ldots, i_N)$ *with* $i_n \in \{1, \ldots, I_n\}$ *for every* $1 \leq n \leq N$.

For example, a third-order tensor (or three way array) has three modes (or indices or dimensions) as shown in figure 3.1 :

In the next figure 3.2 we give an illustration of tensors with different orders :

Definition 3.1.2. ***Fibers***

The fibers are the individual components of a tensor along a specific mode, defined by fixing every index except one.

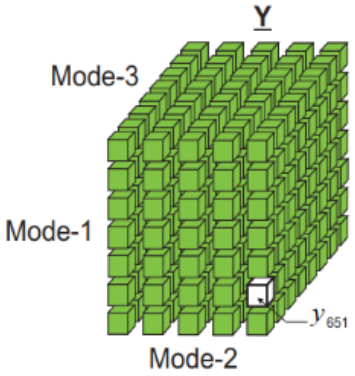

FIGURE 3.1 – Illustration of a third-order tensor $\mathcal{Y} \in \mathbb{R}^{7\times5\times8}$

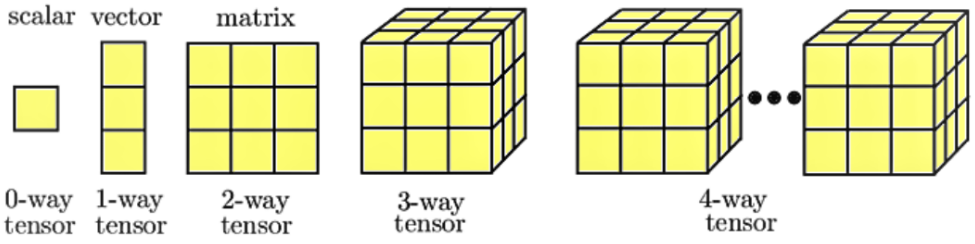

FIGURE 3.2 – Illustration of multi-way data

Definition 3.1.3. ***Slices***

The slices are two-dimensional sections of a tensor, defined by fixing all indices except two.

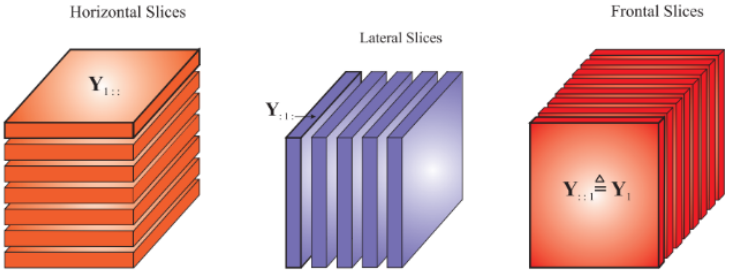

FIGURE 3.4 – Slices : for a third-order tensor

Now we will give short definitions of cubical, symmetric and diagonal tensor :

Definition 3.1.4. ***Cubical tensor***

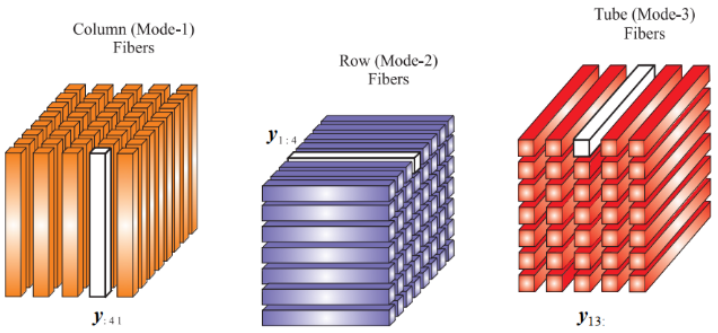

FIGURE 3.3 – Fibers : for a third-order tensors

A tensor $\mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is called a cubical tensor if every mode is the same size, i.e., $I_1 = I_2 = \cdots = I_N$.

Definition 3.1.5. ***Symmetric tensor***

A tensor $\mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is symmetric if $y_{i_1 i_2 \ldots i_N} = y_{p(i_1) p(i_2) \ldots p(i_N)}$ over any permutation p. Here, a tensor $\mathcal{Y} \in \mathbb{R}^{d \times d \times \cdots \times d}$ is said to be symmetric if $y_{i,j,k} = y_{j,i,k} = y_{j,k,i} = y_{k,j,i} = y_{k,i,j}$ for all $1 \leq i, j, k \leq d$.

Definition 3.1.6. ***Diagonal tensor***

A tensor $\mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is diagonal if its elements $y_{i_1 \ldots i_N} \neq 0$ only if $i_1 = \cdots = i_N$.
We use $\mathcal{I}$ to denote the identity tensor with ones on the super-diagonal.

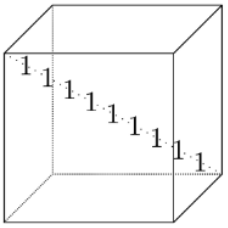

FIGURE 3.5 – three-way tensor of size $\mathcal{I} \times \mathcal{I} \times \mathcal{I}$ with ones along the superdiagonal.

3.1.2 Tensor operations

In order to work with tensors, it is often convenient to transform them to vectors or matrices using the process of vectorisation or matricization.

Definition 3.1.7.

Consider the following transformation $\Psi : \mathbb{T}_{j_1 \dots j_N k_1 \dots k_M} \rightarrow \mathbb{M}_{|J|,|K|}$ *with* $\Psi(\mathcal{A}) = A$ *defined component-wise as*

$$A_{j_1 \dots j_N k_1 \dots k_M} \rightarrow A_{i_{vec}(j,J), i_{vec}(k,K)},$$

where we refer to $J = \{J_1, \dots, J_N\}$, $K = \{K_1, \dots, K_M\}$ *and* $\mathbb{T}_{j_1 \dots j_N k_1 \dots k_M}$ *is the set of all tensors in* $\mathbb{R}^{J_1 \times \dots \times J_N \times K_1 \times \dots \times K_M}$. $\mathbb{M}_{|J|,|K|}$ *is the set of matrices in* $\mathbb{R}^{|J| \times |K|}$ *where* $|J| = J_1 \cdots J_N$ *and* $|K| = K_1 \cdots K_N$. *The index mapping ivec* $(\cdot, \cdot)$ *introduced in [] is given by*

$$i_{vec}(j, J) = j_1 + \sum_{l=2}^{N} (j_l - 1) \prod_{s=1}^{l-1} J_s,$$

$$i_{vec}(k, K) = k_1 + \sum_{l=2}^{M} (k_l - 1) \prod_{s=1}^{l-1} K_s.$$

Definition 3.1.8. ***Vectorisation***

The vectorization of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ *is denoted by* $\mathrm{vec}(\mathcal{X})$, *where*

$$vec \ : \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N} \rightarrow \mathbb{R}^{I_1 . I_2 \dots I_N}$$

stacks the entries of a tensor in reverse lexicographical order into a long column vector.

Example 3.1.1. *Let* $\mathcal{X} \in \mathbb{R}^{2 \times 2 \times 2}$ *be a third order tensor such that :*

$$\mathcal{X}(:,:,1) = \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix}, \quad \mathcal{X}(:,:,2) = \begin{pmatrix} 5 & 7 \\ 6 & 8 \end{pmatrix} \Rightarrow \mathrm{vec}(\mathcal{X}) = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \end{pmatrix}$$

Definition 3.1.9. ***Matricization (n-mode unfolding)***

The n-mode unfolding of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ is denoted by $\mathcal{X}_{(n)}$. It arranges the n-mode fibers to be the columns of the resulting matrix, with $\mathcal{X}_{(n)} \in \mathbb{R}^{I_n \times \prod_{i=1, i \neq n}^{N} I_i}$, every row of $\mathcal{X}_{(n)}$ keeps constant i_n of all components. We have then :

$$\left(\mathcal{X}_{(n)}\right)_{i_n, j} = \mathcal{X}_{i_1, \ldots, i_N}, \text{ where } j = 1 + \sum_{k=1, k \neq n}^{N} (i_k - 1) J_k \text{ and } J_k = \prod_{m=1, m \neq n}^{k-1} I_m.$$

Observe that in the n-mode unfolding the n-mode fibers are rearranged to be the columns of the matrix $\mathcal{X}_{(n)}$.

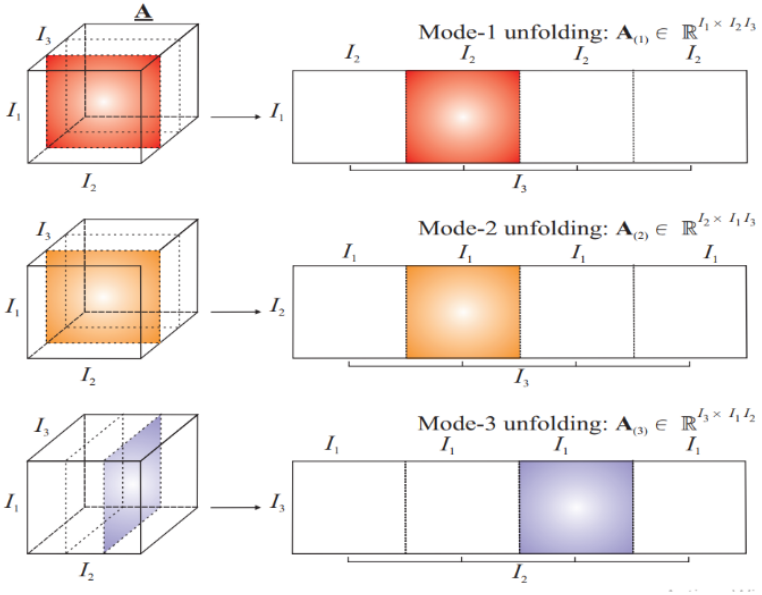

FIGURE 3.6 – Illustration of unfolding a third-order tensor into a matrix in three modes

Example 3.1.2. *Let the frontal slices of $\mathcal{X} \in \mathbb{R}^{3 \times 4 \times 2}$ be :*

$$\mathcal{X}(:,:,1) = \begin{pmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{pmatrix}, \mathcal{X}(:,:,2) = \begin{pmatrix} 13 & 16 & 19 & 22 \\ 14 & 17 & 20 & 23 \\ 15 & 18 & 21 & 24 \end{pmatrix}$$

Then the three mode-n unfolding of X are :

$$\mathcal{X}_{(1)=}\begin{pmatrix} 1 & 4 & 7 & 10 & 13 & 16 & 19 & 22 \\ 2 & 5 & 8 & 11 & 14 & 17 & 20 & 23 \\ 3 & 6 & 9 & 12 & 15 & 18 & 21 & 23 \end{pmatrix}, \mathcal{X}_{(2)} = \begin{pmatrix} 1 & 2 & 3 & 13 & 14 & 15 \\ 4 & 5 & 6 & 16 & 17 & 18 \\ 7 & 8 & 9 & 19 & 20 & 21 \\ 10 & 11 & 12 & 22 & 23 & 24 \end{pmatrix}$$

$$\mathcal{X}_{(3)} = \begin{pmatrix} 1 & 2 & 3 & 4 & \cdots & 8 & 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 & \cdots & 20 & 21 & 22 & 23 & 24 \end{pmatrix}.$$

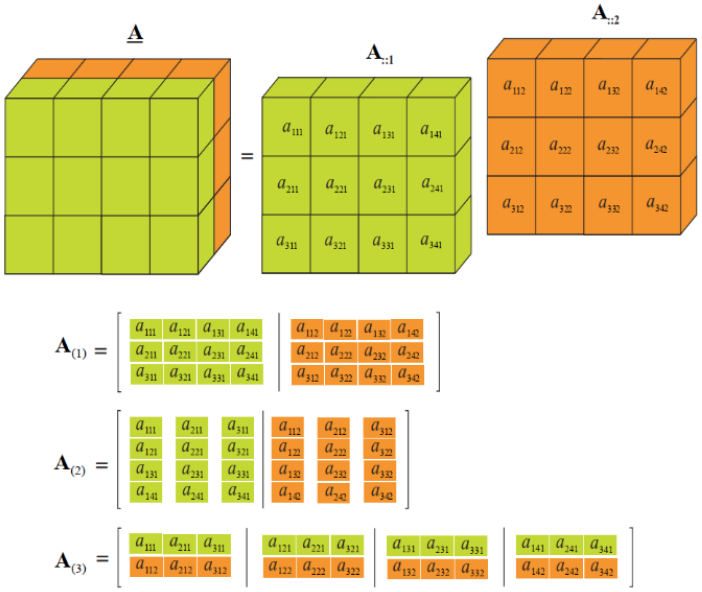

FIGURE 3.7 – Example of unfolding the third-order tensor in mode-1, mode-2 and mode-3

Remark 3.1.1. *Let $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ a 3^{rd}-mode tensor. We can express mode- n unfolding using slices*

$$\mathcal{X}_{(1)} = (\mathcal{X}(:,:,1) \quad \mathcal{X}(:,:,2) \quad \mathcal{X}(:,:,3) \quad \ldots \mathcal{X}(:,:,I_3))$$
$$\mathcal{X}_{(2)} = \left(\mathcal{X}(:,:,1)^\top \mathcal{X}(:,:,2)^\top \mathcal{X}(:,:,3)^\top \ldots \mathcal{X}(:,:,I_3)^\top\right)$$
$$\mathcal{X}_{(3)} = \left(\mathcal{X}(:,1,:)^\top \mathcal{X}(:,2,:)^\top \mathcal{X}(:,3,:)^\top \ldots \mathcal{X}(:,I_2,:)^\top\right).$$

An important operation for a tensor is the tensor-matrix multiplication, also known as n-mode product of a tensor, given by the definition below :

Definition 3.1.10. ***The n-mode product***

The n-mode product of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ with a matrix $U \in \mathbb{R}^{J \times I_n}$ is denoted by $\mathcal{X} \times_n U$ of size $I_1 \times \ldots \times I_{n-1} \times J \times I_{n+1} \times \ldots \times I_N$, its elements are given by :

$$(\mathcal{X} \times_n U)_{i_1,\ldots,i_{n-1},j,i_{n+1},\ldots,i_N} = \sum_{i_n=1}^{I_n} \mathcal{X}_{i_1,\ldots,i_{n-1},i_n,i_{n+1},\ldots,i_N} U_{j,i_n}$$

Each mode-n fiber is multiplied by the matrix U.

Example 3.1.3. *Let the frontal slices of $\mathcal{X} \in \mathbb{R}^{3\times4\times2}$ and $U \in \mathbb{R}^{2\times3}$ be :*

$$\mathcal{X}(:,:,1) = \begin{pmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{pmatrix}, \mathcal{X}(:,:,2) = \begin{pmatrix} 13 & 16 & 19 & 22 \\ 14 & 17 & 20 & 23 \\ 15 & 18 & 21 & 24 \end{pmatrix}, U = \begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$$

Then the mode- 1 product of $\mathcal{X}$ and U is denoted by $\mathcal{Y} = \mathcal{X} \times_1 U \in \mathbb{R}^{2\times4\times2}$, and the result is :

$$\mathcal{Y}(:,:,1) = \begin{pmatrix} 22 & 49 & 76 & 103 \\ 28 & 64 & 100 & 136 \end{pmatrix}, \mathcal{Y}(:,:,2) = \begin{pmatrix} 130 & 157 & 184 & 211 \\ 172 & 208 & 244 & 280 \end{pmatrix}$$

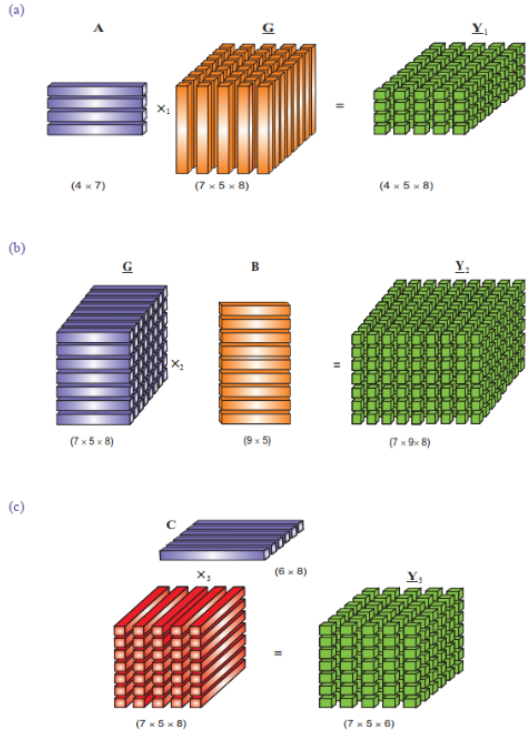

FIGURE 3.8 – llustration of the mode-n multiplications of a third-order tensor by matrices. (a) mode-1 multiplication. (b) mode-2 multiplication. (c) mode-3 multiplication.

Proposition 3.1.1. *Given a tensors* $\mathcal{X} \in \mathbb{R}^{I_1 \times \ldots \times I_{n-1} \times I_n \times I_{n+1} \times \ldots \times I_N}$*, and a matrices* $U \in \mathbb{R}^{J \times I_n}$*,* $\mathcal{Y} = \mathcal{X} \times {}_n U \in \mathbb{R}^{I_1 \times \ldots \times I_{n-1} \times J \times I_{n+1} \times \ldots \times I_N}$ *and we have :*

$$\mathcal{Y} = \mathcal{X} \times_n U \iff \mathcal{Y}_{(n)} = U\mathcal{X}_{(n)}.$$

Proof 3.1.1.

$$(\mathcal{X} \times_n U)_{i_1,..,i_{n-1},j,i_{n+1},\ldots,i_N} = \sum_{i_n=1}^{I_n} \mathcal{X}_{i_1,\ldots,i_{n-1},i_n,i_{n+1},\ldots,i_N} U_{ji_n} = \mathcal{Y}_{i_1,\ldots,i_{n-1},j,i_{n+1},\ldots,i_N}.$$

So each mode-n fiber of $\mathcal{X}$ *is multiplied by the matrix* U *:*

$$\mathcal{Y}_{i_1,\ldots,i_{n-1},:,i_{n+1},\ldots,i_N} = U\mathcal{X}_{i_1,\ldots,i_{n-1},:,i_{n+1},\ldots,i_N}.$$

Thus :

$$\mathcal{Y}_{(n)} = U\mathcal{X}_{(n)}$$

Proposition 3.1.2.

— *For distinct modes in a series of multiplication, the order of the multiplications is irrelevant*

$$(\mathcal{X} \times_n U) \times_m V = (\mathcal{X} \times_m V) \times_n U = \mathcal{X} \times_n U \times_m V, \text{ where } n \neq m$$

— *If the modes are the same, then*

$$(\mathcal{X} \times_n U) \times_n V = \mathcal{X} \times_n VU.$$

— *Let $A \in \mathbb{R}^{I_1 \times I_2}, U \in \mathbb{R}^{J \times I_1}$ and $V \in \mathbb{R}^{K \times I_2}$ be three matrices, we have*

$$A \times_1 U = UA$$
$$A \times_2 V = AV^\top.$$

Definition 3.1.11. ***The n-mode vector product***

Let $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ be an N-th order tensor and $v \in \mathbb{R}^{I_n}$ be a vector. The n-mode product of $\mathcal{X}$ by v denoted by $\mathcal{X}\overline{\times}_n v$ is a tensor of size $I_1 \times \ldots \times I_{n-1} \times I_{n+1} \times \ldots \times I_N$ (of order $N-1$) and whose entries are given by :

$$(\mathcal{X}\overline{\times}_n v)_{i_1,\ldots,i_{n-1},i_{n+1},\ldots,i_N} = \sum_{i_n=1}^{I_n} \mathcal{X}_{i_1,\ldots,i_{n-1},i_n,i_{n+1},\ldots,i_N} v_{i_n}$$

The idea is to compute the inner product of each mode-n fiber with the vector **v**.

Example 3.1.4. *Let the frontal slices of $\mathcal{X} \in \mathbb{R}^{3 \times 4 \times 2}$ be :*

$$\mathcal{X}(:,:,1) = \begin{pmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{pmatrix}, \mathcal{X}(:,:,2) = \begin{pmatrix} 13 & 16 & 19 & 22 \\ 14 & 17 & 20 & 23 \\ 15 & 18 & 21 & 24 \end{pmatrix}$$

and define $\mathbf{v} = [1\ 2\ 3\ 4]^\top$. *Then*

$$\mathcal{X} \times_2 \mathbf{v} = \begin{bmatrix} 70 & 190 \\ 80 & 200 \\ 90 & 210 \end{bmatrix}.$$

When it comes to mode-n vector multiplication, precedence matters because the order of the intermediate results changes. In other words,

$$\mathcal{X} \times_m \times_n \mathbf{b} = (\mathcal{X} \times_m a) \times_{n-1} \mathbf{b} = (\mathcal{X} \times_n \mathbf{b}) \times_m a \quad \text{for} \quad m < n.$$

Definition 3.1.12. ***The outer product***

The outer product of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times .. \times I_N}$ and $\mathcal{Y} \in \mathbb{R}^{J_1 \times J_2 ... \times J_M}$ is a tensor denoted by $\mathcal{X} \circ \mathcal{Y} = Z \in \mathbb{R}^{I_1 \times I_2 \times ... \times I_N \times J_1 \times J_2 \times ... \times J_M}$, its elements are given by :

$$\mathcal{Z}_{i_1,i_2,...,i_N,j_1,j_2,...,j_M} = \mathcal{X}_{i_1,i_2,...,i_N} \mathcal{Y}_{j_1,j_2,...,j_M}.$$

Observe that, the tensor $\mathcal{Z}$ contains all the possible combinations of pair-wise products between the elements of $\mathcal{Y}$ and $\mathcal{X}$.

Remark 3.1.2. *If $v_1, v_2, \ldots, v_N$ are N vectors of sizes I_i for $i = 1, \ldots, N$, their outer product is an N^{th} -order tensor of size $I_1 \times \ldots \times I_N$ and we have :*

$$(v_1 \circ v_2 \circ \ldots \circ v_N)_{i_1,i_2,...,i_N} = v_1(i_1)\, v_2(i_2) \ldots v_N(i_N)$$

For the matrix case we know that the rank of a matrix is the number of linearly independent column vectors, or, equivalently, the number of non-zero singular values. However, this definition of the matrix rank is not directly extensible to tensors. In order to extend the notion of rank to tensors, it has to be considered from a different perspective.

Definition 3.1.13. ***Rank-one tensor***

An N-way tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 ... \times I_N}$ is rank one if it can be written as the outer product of N vectors. That means $\mathcal{X} = x^{(1)} \circ x^{(2)} \circ \ldots \circ x^{(N)}$, where

$$\mathcal{X}_{i_1,i_2,...,i_N} = \prod_{k=1}^{N} x_{i_k}^{(k)} \text{ for all } 1 \leq i_k \leq I_N$$

and $x_{i_k}^{(k)}$ denotes the i_k^{th} element of vector $x_k \in \mathbb{R}^{I_k}$.
A tensor is of rank $R \in \mathbb{N}$ if it could be written as the sum of R rank one tensors.

Remark 3.1.3. *As special cases, the outer product of two vectors $u \in \mathbb{R}^I$ and $v \in \mathbb{R}^J$ yields a rank-one matrix :*

$$U = u \circ v = uv^\top \in \mathbb{R}^{I \times J}$$

and the outer product of three vectors $a \in \mathbb{R}^I, b \in \mathbb{R}^J$ *and* $c \in \mathbb{R}^Q$ *yields a third-order rank-one tensor :*

$$\mathcal{X} = a \circ b \circ c \in \mathbb{R}^{I \times J \times Q},$$

where

$$\mathcal{X}_{i,j,q} = a_i b_j c_q$$

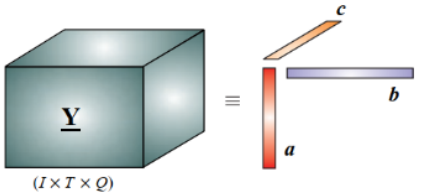

FIGURE 3.9 – Illustration of a rank-one third-order tensor

Definition 3.1.14. ***The transpose of a tensor***

Let $\mathcal{A} \in \mathbb{R}^{I_1 \times \ldots \times I_N \times K_1 \times \ldots \times K_M}$ *and* $\mathcal{B} \in \mathbb{R}^{K_1 \times \ldots \times K_M \times I_1 \times \ldots \times I_N}$ *be two tensors verified* $b_{k_1 \ldots k_M i_1 \ldots i_N} = a_{i_1 \ldots i_N k_1 \ldots k_M}$, *then* $\mathcal{B}$ *is called the* transpose *of* $\mathcal{A}$ *and denoted by* $\mathcal{A}^\top$.

Definition 3.1.15. ***The Kronecker product***

The Kronecker product of two tensors $\mathcal{X} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ *and* $\mathcal{Y} \in \mathbb{R}^{J_1 \times \ldots \times J_N}$ *is defined by*

$$\mathcal{Z} = \mathcal{X} \otimes \mathcal{Y} \in \mathbb{R}^{I_1 J_1 \times \ldots \times I_N J_N}$$

where

$$\mathcal{Z}_{i_1 j_1, \ldots, i_N j_N} = \mathcal{X}_{i_1, \ldots, i_N} \mathcal{Y}_{j_1, \ldots, j_N}, \quad \text{for } i_n = 1, \ldots, I_n, \quad j_n = 1, \ldots, J_n, \quad n = 1, \ldots, N$$

Remark 3.1.4. *The Kronecker product can be viewed as a form of vectorization*

(or flattening) of the outer product. In particular, for two column vectors a and b, we can write :

$$a \otimes b = \mathrm{vec}(b \circ a).$$

Another similar identity that further highlights the similarity between the operations is :

$$a \otimes b^\top = ab^\top = a \circ b.$$

3.1.3 T-product

Discrete Fourier Transformation

The Discrete Fourier Transformation (DFT) plays a very important role in the definition of the T-product of tensors. The DFT on a vector $v \in \mathbb{R}^n$ is defined by

$$\tilde{v} = F_n v \in \mathbb{C}^n, \tag{3.1}$$

where F_n is the matrix defined as

$$F_n = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{n-1} & \omega^{2(n-1)} & \cdots & \omega^{(n-1)(n-1)} \end{pmatrix} \in \mathbb{C}^{n \times n}, \tag{3.2}$$

where $\omega = e^{-\frac{2\pi i}{n}}$ with $i^2 = -1$. It is not difficult to show that :

$$F_n^* = \overline{F}_n, \quad \text{and} \quad F_n^* F_n = F_n F_n^* = nI_n. \tag{3.3}$$

Then $F_n^{-1} = \frac{1}{n} F_n$ which shows that $\frac{1}{\sqrt{n}} F_n$ is a unitary matrix. The cost of computing the vector $\tilde{v}$ directly from (3.1) is $O(n^2)$. Using the FFT, it will cost only $O(n \log(n))$ and this makes the FFT very fast for large problems. It is known that

$$F_n \mathbf{circ}(v) F_n^{-1} = \mathbf{Diag}(\tilde{v}), \tag{3.4}$$

which is equivalent to

$$F_n \mathbf{circ}(v) F_n^* = n \mathbf{Diag}(\tilde{v}), \tag{3.5}$$

where

$$\mathbf{circ}(v) = \begin{pmatrix} v_1 & v_2 & \cdots & v_n \\ v_2 & v_1 & \cdots & v_{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ v_n & v_{n-1} & \cdots & v_1 \end{pmatrix},$$

and $\mathrm{Diag}(\tilde{v})$, is the diagonal matrix whose i-th diagonal element is $\tilde{v}_i$.

Definitions and properties of the T-product

In this part, we briefly review some concepts and notations related to the T-Product. Let $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ be a third-order tensor, then the operations bcirc, unfold and fold are defined by

$$\mathrm{bcirc}(\mathcal{A}) = \begin{pmatrix} A_1 & A_{n_3} & A_{n_3-1} & \dots & A_2 \\ A_2 & A_1 & A_{n_3} & \dots & A_3 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ A_{n_3} & A_{n_3-1} & \ddots & A_2 & A_1 \end{pmatrix} \in \mathbb{R}^{n_1 n_3 \times n_2 n_3},$$

$$\mathrm{unfold}(\mathcal{A}) = \begin{pmatrix} A_1 \\ A_2 \\ \vdots \\ A_{n_3} \end{pmatrix} \in \mathbb{R}^{n_1 n_3 \times n_2}, \quad \textbf{fold}\ (\mathrm{unfold}(\mathcal{A})) = \mathcal{A}$$

Let $\widetilde{\mathcal{A}}$ be the tensor obtained by applying the DFT on all the 3-mode tubes of the tensor $\mathcal{A}$. With the Matlab command fft, we have

$$\widetilde{\mathcal{A}} = \mathrm{fft}(\mathcal{A}, [], 3), \text{ and } \mathrm{ifft}(\widetilde{\mathcal{A}}, [], 3) = \mathcal{A},$$

where ifft denotes the inverse fast fourier transform.
The tensor $\tilde{A}$ can be obtained using the 3 -mode product as follows

$$\widetilde{\mathcal{A}} = \mathcal{A} \times_3 F_{n_3}, \quad \mathcal{A} = \widetilde{\mathcal{A}} \times {}_3 F_{n_3}^{-1}$$

Let A be the matrix

$$\mathbf{A} = \text{BlockDiag}(\widetilde{\mathcal{A}}) = \begin{pmatrix} A^{(1)} & & & \\ & A^{(2)} & & \\ & & \ddots & \\ & & & A^{(n_3)} \end{pmatrix}$$

and the matrices $A^{(i)}$'s are the frontal slices of the tensor $\tilde{\mathcal{A}}$. The block circulant matrix bcirc $(\mathcal{A})$ can also be block diagonalized by using the DFT and this gives

$$(F_{n_3} \otimes I_{n_1}) \,\text{bcirc}(\mathcal{A}) \left(F_{n_3}^* \otimes I_{n_2}\right) = \mathbf{A}$$

The diagonal blocks of the matrix A satisfy the following property

$$\begin{cases} A^{(1)} \in \mathbb{R}^{n_1 \times n_2} \\ \text{conj}\left(A^{(i)}\right) = A^{(n_3 - i + 2)} \end{cases}$$

where $\text{conj}\left(A^{(i)}\right)$ is the complex conjugate of the matrix $A^{(i)}$. Next we recall the definition of the T-product.

Definition 3.1.16.

The T-product ($\star$) between two tensors $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and $\mathcal{B} \in \mathbb{R}^{n_2 \times m \times n_3}$ is the $n_1 \times m \times n_3$ tensor given by :

$$\mathcal{A} \star \mathcal{B} = \textit{fold}\,(\text{bcirc}(\mathcal{A})\,\text{unfold}(\mathcal{B})).$$

The following algorithm allows us to compute in an efficient way the T-product of the tensors $\mathcal{A}$ and $\mathcal{B}$.

Algorithm 6 Computing the T-product via FFT

Require: $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and $\mathcal{B} \in \mathbb{R}^{n_2 \times m \times n_3}$

Ensure: $\mathcal{C} = \mathcal{A} \star \mathcal{B} \in \mathbb{R}^{n_1 \times m \times n_3}$

Compute $\tilde{\mathcal{A}} = \text{fft}(\mathcal{A}, [], 3)$ and $\tilde{\mathcal{B}} = \text{fft}(\mathcal{B}, [], 3)$.

Compute each frontal slice of $\tilde{\mathcal{C}}$ by

$$\mathcal{C}^{(i)} = \begin{cases} \tilde{\mathcal{A}}^{(i)} \tilde{\mathcal{B}}^{(i)}, & \text{for } i = 1, \ldots, \left\lfloor \frac{n_3+1}{2} \right\rfloor \\ \text{conj}(\mathcal{C}^{(n_3+2-i)}), & \text{for } i = \left\lfloor \frac{n_3+1}{2} \right\rfloor + 1, \ldots, n_3 \end{cases}$$

Compute $\mathcal{C} = \text{ifft}(\tilde{\mathcal{C}}, [], 3)$.

For the T-product, we have the following definitions

Definition 3.1.17.

1. *The identity tensor $\mathcal{J}_{n_1 n_1 n_3}$ is the tensor whose first frontal slice is the identity matrix $I_{n_1 n_1}$ and the other frontal slices are all zeros.*
2. *An $n_1 \times n_1 \times n_3$ tensor $\mathcal{A}$ is invertible, if there exists a tensor $\mathcal{B}$ of order $n_1 \times n_1 \times n_3$ such that*

$$\mathcal{A} \star \mathcal{B} = \mathcal{J}_{\mathfrak{n}_1 n_1 n_3} \quad \text{and} \quad \mathcal{B} \star \mathcal{A} = \mathcal{J}_{n_1 n_1 n_3}$$

In that case, we set $\mathcal{B} = \mathcal{A}^{-1}$. It is clear that $\mathcal{A}$ is invertible if and only if $\mathrm{bcirc}(\mathcal{A})$ *is invertible.*

3. *The transpose of $\mathcal{A}$ is obtained by transposing each of the frontal slices and then reversing the order of transposed frontal slices 2 through n_3.*
4. *If $\mathcal{A}, \mathcal{B}$ and $\mathcal{C}$ are tensors of appropriate order, then*

$$(\mathcal{A} \star \mathcal{B}) \star \mathcal{C} = \mathcal{A} \star (\mathcal{B} \star \mathcal{C})$$

5. *Suppose $\mathcal{A}$ and $\mathcal{B}$ are two tensors such $\mathcal{A} \star \mathcal{B}$ and $\mathcal{B}^T \star \mathcal{A}^T$ are defined. Then*

$$(\mathcal{A} \star \mathcal{B})^T = \mathcal{B}^T \star \mathcal{A}^T$$

Definition 3.1.18.

Let $\mathcal{A}$ and $\mathcal{B}$ two tensors in $\mathbb{R}^{n_1 \times n_2 \times n_3}$. Then

1. *The scalar inner product is defined by*

$$\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} \sum_{i_3=1}^{n_3} a_{i_1 i_2 i_3} b_{i_1 i_2 i_3}.$$

2. *The norm of $\mathcal{A}$ is defined by*

$$\|\mathcal{A}\|_F = \sqrt{\langle \mathcal{A}, \mathcal{A} \rangle}$$

Remark 3.1.5. *Another interesting way for computing the scalar product and*

the associated norm is as follows :

$$\langle \mathcal{A}, \mathcal{B} \rangle = \frac{1}{n_3} \langle \mathbf{A}, \mathbf{b} \rangle ; \|\mathcal{A}\|_F = \frac{1}{\sqrt{n_3}} \|\mathbf{A}\|_F$$

where the block diagonal matrix $\mathbf{A}$ *is defined by (3.2).*

Definition 3.1.19.

1. *An* $n_1 \times n_1 \times n_3$ *tensor* Q *is orthogonal if*

$$Q^T \star Q = Q \star Q^T = \mathcal{J}_{n_1 n_1 n_3}.$$

2. *A tensor is called f-diagonal if its frontal slices are orthogonal matrices. It is called upper triangular if all its frontal slices are upper triangular.*

Definition 3.1.20. ***Block tensor based on T-product***

Suppose $\mathcal{A} \in \mathbb{R}^{n_1 \times m_1 \times n_3}, \mathcal{B} \in \mathbb{R}^{n_1 \times m_2 \times n_3}, \mathcal{C} \in \mathbb{R}^{n_2 \times m_1 \times n_3}$ *and* $\mathcal{D} \in \mathbb{R}^{n_2 \times m_2 \times n_3}$ *are four tensors. The block tensor*

$$\begin{bmatrix} \mathcal{A} & \mathcal{B} \\ \mathcal{C} & \mathcal{D} \end{bmatrix} \in \mathbb{R}^{(n_1+n_2) \times (m_1+m_2) \times n_3}$$

is defined by compositing the frontal slices of the four tensors.

Proposition 3.7. Let $\mathcal{A}, \mathcal{A}_1 \in \mathbb{R}^{n \times s \times n_3}, \mathcal{B}, \mathcal{B}_1 \in \mathbb{R}^{n \times p \times n_3}, \mathcal{A}_2 \in \mathbb{R}^{\ell \times s \times n_3}, \mathcal{B}_2 \in \mathbb{R}^{\ell \times p \times n_3}, \mathcal{C} \in \mathbb{R}^{s \times n \times n_3}, \mathcal{D} \in \mathbb{R}^{p \times n \times n_3}$ and $\mathcal{F} \in \mathbb{R}^{n \times n \times n_3}$. Then

1. $\mathcal{F} \star [\mathcal{A}\mathcal{B}] = [\mathcal{F} \star \mathcal{A} \quad \mathcal{F} \star \mathcal{B}] \in \mathbb{R}^{n \times (s+p) \times n_3}$
2. $\begin{bmatrix} \mathcal{C} \\ \mathcal{D} \end{bmatrix} \star \mathcal{F} = \begin{bmatrix} \mathcal{C} \star \mathcal{F} \\ \mathcal{D} \star \mathcal{F} \end{bmatrix} \in \mathbb{R}^{(s+p) \times n \times n_3}$
3. $[\mathcal{A}\mathcal{B}] \star \begin{bmatrix} \mathcal{C} \\ \mathcal{D} \end{bmatrix} = \mathcal{A} \star \mathcal{C} + \mathcal{B} \star \mathcal{D} \in \mathbb{R}^{n \times n \times n_3}$
4. $\begin{bmatrix} \mathcal{A}_1 & \mathcal{B}_1 \\ \mathcal{A}_2 & \mathcal{B}_2 \end{bmatrix} \star \begin{bmatrix} \mathcal{C} \\ \mathcal{D} \end{bmatrix} = \begin{bmatrix} \mathcal{A}_1 \star \mathcal{C} + \mathcal{B}_1 \star \mathcal{D} \\ \mathcal{A}_2 \star \mathcal{C} + \mathcal{B}_2 \star \mathcal{D} \end{bmatrix} \in \mathbb{R}^{(\ell+n) \times n \times n_3}$

3.1.4 Einstein product

The tensor Einstein product is a multidimensional generalizations of matrix product :

Definition 3.1.21. ***Einstein product***

Let $\mathcal{A} \in \mathbb{R}^{I_1 \times ... \times I_l \times K_1 \times ... \times K_N}$, $\mathcal{B} \in \mathbb{R}^{K_1 \times ... \times K_N \times J_1 \times ... \times J_M}$, the Einstein product of the tensors $\mathcal{A}$ and $\mathcal{B}$ is the tensor of size $(I_1 \times ... \times I_l \times J_1 \times ... \times J_M)$ defined by

$$(\mathcal{A} *_N \mathcal{B})_{i_1 ... i_l j_1 ... j_M} = \sum_{k_1=1}^{K_1} \sum_{k_2=1}^{K_2} ... \sum_{k_N=1}^{K_N} a_{i_1 ... i_l k_1 ... k_N} b_{k_1 ... k_N j_1 ... j_M}$$

A special case of the Einstein product by assuming that $N = 1$, called The contract product :

Definition 3.1.22. ***Contract product***

Let $\mathcal{X} \in \mathbb{R}^{I_1 \times ... \times I_L}$ and $\mathcal{Y} \in \mathbb{R}^{J_1 \times ... \times J_M}$ with $I_L = J_1$, the contract product of $\mathcal{X}$ by $\mathcal{Y}$ is given by $\mathcal{Z} = \mathcal{X} *_N \mathcal{Y} \in \mathbb{R}^{I_1 \times ... \times I_{N-1} \times J_2 \times ... \times J_M}$, its elements are given by

$$\mathcal{Z}_{i_1,...,i_{N-1},j_2,...,j_M} = \sum_{i_N=1}^{I_N} \mathcal{X}_{i_1,...,i_N} \mathcal{Y}_{i_N,j_2,...,j_M}$$

The inverse of a square tensor is defined as follows.

Definition 3.1.23. ***Inverse of a tensor***

A square tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times ... \times I_N \times I_1 \times ... \times I_N}$ is invertible iff there exists a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times ... \times I_N \times I_1 \times ... \times I_N}$ such that

$$\mathcal{A} *_N \mathcal{X} = \mathcal{X} *_N \mathcal{A} = \mathcal{I} \in \mathbb{R}^{I_1 \times ... \times I_N \times I_1 \times ... \times I_N}.$$

In that case, $\mathcal{X}$ is the inverse of $\mathcal{A}$ and is denoted by $\mathcal{A}^{-1}$.

Proposition 3.1.3. *Consider $\mathcal{A} \in \mathbb{R}^{I_1 \times ... \times I_N \times K_1 \times ... \times K_M}$ and $\mathcal{B} \in \mathbb{R}^{K_1 \times ... \times K_M \times I_1 \times ... \times I_N}$. Then we have the following relations*

1. $(\mathcal{A} *_M \mathcal{B})^\top = \mathcal{B}^\top *_N \mathcal{A}^\top$.
2. $\mathcal{I}_M *_M \mathcal{B} = \mathcal{B}$ *and* $\mathcal{B} *_N \mathcal{I}_N = \mathcal{B}$.

The *trace* denoted by $\text{tr}(\cdot)$ of a square-order tensor $\mathcal{A} \in \mathbb{R}^{J_1 \times \cdots \times J_N \times J_1 \times \cdots \times J_N}$ is given by

$$\text{tr}(\mathcal{A}) = \sum_{j_1,...,j_N} a_{j_1 ... j_N j_1 ... j_N}.$$

The inner product of the two tensors $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{J_1 \times \cdots \times J_N \times K_1 \times \cdots \times K_M}$ is defined as

$$\langle \mathcal{X}, \mathcal{Y} \rangle = \mathbf{tr}(\mathcal{X}^T *_N \mathcal{Y}),$$

where $\mathcal{Y}^T \in \mathbb{R}^{K_1 \times \cdots \times K_M \times J_1 \times \cdots \times J_N}$ is the transpose tensor of $\mathcal{Y}$. If $\langle \mathcal{X}, \mathcal{Y} \rangle \geq 0$, we say that $\mathcal{X}$ and $\mathcal{Y}$ are orthogonal. The corresponding norm is the tensor Frobenius norm given by

$$\|\mathcal{X}\| = \sqrt{\mathbf{tr}(\mathcal{X}^T *_N \mathcal{X})}.$$

3.1.5 Block Tensor

In this subsection, based on [10] and analogously to block matrices, we describe briefly how our tensors can be stored properly in a one tensor. To clarify our description we need to start by giving the definition of an even-order paired tensor.

Definition 3.1.24. ***even-order paired tensor***

A tensor is called even-order paired tensor if its order is always even, i.e., ($2\,\mathrm{N}$ *for example), and the elements of the tensor are indicated using a pairwise index notation, i.e.,* $a_{j_1 i_1 \ldots j_N i_N}$ *for* $\mathcal{A} \in \mathbb{R}^{J_1 \times I_1 \times \ldots \times J_N \times I_N}$.

A definition of n-mode row block tensor, of two even-order paired tensor of the same size is given by

Definition 3.1.25. ***n-mode row block tensor***

Consider $\mathcal{A}, \mathcal{B} \in \mathbb{R}^{I_1 \times I_1 \times \ldots \times J_N \times I_N}$ *two even-order paired tensors, the* n*-mode row block tensor denoted by* $|\mathcal{AB}|_n \in \mathbb{R}^{I_1 \times I_1 \times \ldots \times I_n \times 2I_n \times \ldots \times I_N \times I_N}$ *is defined by the following*

$$(|\mathcal{AB}|_n)_{j_1 \ldots j_N i_1 \ldots i_N} = \begin{cases} a_{j_1 \ldots j_N i_1 \ldots i_N}, & j_k = 1, \ldots, J_k, i_k = 1, \ldots, I_k \forall k \\ b_{j_1 \ldots j_N i_1 \ldots i_N}, & j_k = 1, \ldots, J_k, \forall k, i_k = 1, \ldots, I_k \textit{ for } k \neq n \\ & \textit{and } i_k = I_k + 1, \ldots, 2I_k \textit{ for } k = n \end{cases}$$

In this work, we are interested in the 4 th-order tensors (i.e., $N = 2$), and not especially paired ones (i.e., the paired wise index notation is not used). We

have noticed that the definition above is still valid for the case of 4 th-order paired or non-paired tensor. In this regard, we specify more in the following this block tensor notation by giving some examples. We suppose now that $\mathcal{A}$ and $\mathcal{B}$ are two $4th$-order tensors in the space $\mathbb{R}^{J_1\times J_2\times I_1\times I_2}$. By following the two definitions above, we refer to 1 -mode row block tensor by $|\mathcal{AB}|_1 \in \mathbb{R}^{J_1\times 2J_2\times I_1\times I_2}$. We use the MATLAB colon operation : to explain how to extract the two tensors $\mathcal{A}$ and $\mathcal{B}$ as follows

$$|\mathcal{AB}|_1 (:, 1:J_2, :::) = \mathcal{A} \quad \text{and} \quad |\mathcal{AB}|_1 (:, J_2+1:2J_2, :::) = \mathcal{B}$$

The notation of 2-mode row block tensor $|\mathcal{AB}|_2 \in \mathbb{R}^{I_1\times J_2\times I_1\times 2I_2}$ described below is the one mostly used in this chapter. The extraction goes as

$$|\mathcal{AB}|_2 (:, :, :, 1:K_2) = \mathcal{A} \quad \text{and} \quad |\mathcal{A}B|_2 (:, :, :, K_2+1:2K_2) = \mathcal{B}$$

Remark 3.1.6. *We notice that if we consider $N = 1$ (i.e., $\mathcal{A}$ and $\mathcal{B}$ are now matrices, we refer to them as A and B), then the notation $|\mathcal{AB}|_1 = |\mathcal{AB}|_2 = [A, B] \in \mathbb{R}^{J_1\times 2I_1}$ is now defined by the standard block matrices definition.*

Similarly to the 1 or 2-mode row block tensor, we can define the 1 or 2-mode column block tensor for the 4th-order tensors. Using Matlab notation, $\left|\begin{array}{c}\mathcal{A}\\ \mathcal{B}\end{array}\right|_1$ is the 1-mode column block tensor in $\mathbb{R}^{2I_1\times J_2\times I_1\times I_2,}$ defined by

$$\left|\begin{array}{c}\mathcal{A}\\ \mathcal{B}\end{array}\right|_1 (1:J_1, :, :, :) = \mathcal{A} \text{ and } \left|\begin{array}{c}\mathcal{A}\\ \mathcal{B}\end{array}\right|_1 (J_1+1:2J_1, :, :, :) = \mathcal{B}$$

and $\left|\begin{array}{c}\mathcal{A}\\ \mathcal{B}\end{array}\right|$ is 2-mode column block tensor in $\mathbb{R}^{I_1\times I_2\times 2I_1\times I_2}$ given by

$$\left|\begin{array}{c}\mathcal{A}\\ \mathcal{B}\end{array}\right|_2 (:, :, 1:I_1, :) = \mathcal{A} \text{ and } \left|\begin{array}{c}\mathcal{A}\\ \mathcal{B}\end{array}\right|_2 (:, :, I_1+1:2I_1, :) = \mathcal{B}$$

Proposition 3.1.4. *Consider four tensors of the same size $\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D} \in \mathbb{R}^{K_1\times K_2\times K_1\times K_2}$, then we get*

$$\underbrace{|\mathcal{AB}|_2}_{\in\mathbb{R}^{K_1\times K_2\times K_1\times 2K_2}} * \underbrace{|\mathcal{CD}|_1}_{\in\mathbb{R}^{K_1\times 2K_2\times K_1\times K_2}} = \mathcal{A}*_2\mathcal{C}+\mathcal{B}*_2\mathcal{D}\in\mathbb{R}^{K_1\times K_2\times K_1\times K_2}$$

$$\underbrace{|\mathcal{AB}|_1}_{\in\mathbb{R}^{K_1\times 2K_2\times K_1\times K_2}} * \underbrace{|\mathcal{CD}|_2}_{\in\mathbb{R}^{K_1\times K_2\times K_1\times 2K_2}} = \left|\left|\mathcal{A}*_2\mathcal{CA}*_2\mathcal{D}\right|_2, \left|\mathcal{B}*_2\mathcal{C} \quad \mathcal{B}*_2\mathcal{D}\right|_2\right|_1\in\mathbb{R}^{K_1\times 2K_2\times K_1\times 2K_2}$$

Let us consider that we have m tensors $\mathcal{A}_k\in\mathbb{R}^{I_1\times I_2\times I_1\times I_2}$, we can apply the definition associated with the 2-mode row block tensor in succession to create a one tensor denoted by $\mathcal{A}$ in the space $\mathbb{R}^{J_1\times I_2\times I_1\times mI_2}$ and has the following form

$$\mathcal{A}=|\mathcal{A}_1 \quad \mathcal{A}_2 \quad \dots \quad \mathcal{A}_m|\in\mathbb{R}^{J_1\times J_2\times I_1\times mI_2} \tag{3.6}$$

— If m is even then we consider the following

$$\mathcal{X}_{12}=\underbrace{|\mathcal{A}_1 \quad \mathcal{A}_2|_2}_{\in\mathbb{R}^{J_1\times J_2\times I_1\times 2I_2}}, \quad \mathcal{X}_{34}=\underbrace{|\mathcal{A}_3 \quad \mathcal{A}_4|_2}_{\in\mathbb{R}^{J_1\times J_2\times I_1\times 2I_2}}, \quad \dots, \quad \mathcal{X}_{m-1m}=\underbrace{|\mathcal{A}_{m-1} \quad \mathcal{A}_m|_2}_{\in\mathbb{R}^{J_1\times J_2\times I_1\times 2I_2}},$$

then we stored them two by two as follows

$$\mathcal{X}_{1234}=\underbrace{|\mathcal{X}_{12} \quad \mathcal{X}_{34}|_2}_{\in\mathbb{R}^{J_1\times J_2\times I_1\times 4I_2}}, \quad \dots, \quad \mathcal{X}_{m-3m-2m-1m}=\underbrace{|\mathcal{X}_{m-3m-2} \quad \mathcal{X}_{m-1m}|_2}_{\in\mathbb{R}^{J_1\times J_2\times I_1\times 4I_2}},$$

We keep repeating the process until the last 2-mode row block tensor obtained and simply we presented in the following notation

$$\mathcal{A}=|\mathcal{A}_1 \quad \mathcal{A}_2 \quad \dots \quad \mathcal{A}_m|\in\mathbb{R}^{J_1\times J_2\times I_1\times mI_2} \tag{3.7}$$

— If m is odd, then we first choose $m-1$ tensors and applied the process described above. We denote by $\mathcal{X}$ the obtained tensor in $\mathbb{R}^{J_1\times I_2\times I_1\times(m-1)I_2}$ and then we obtain the desired tensor $\mathcal{A}_m$ given by

$$\mathcal{A}=|\mathcal{X} \quad \mathcal{A}_m|\in\mathbb{R}^{J_1\times J_2\times I_1\times mI_2}$$

An explanation of the MATLAB colon : for the constructed tensor $\mathcal{A}$ goes as follows

$$\mathcal{A}_l = \mathcal{A}\left(:,:,:,(l-1)I_2+1:lI_2\right) \qquad \text{for } l = 1,\ldots,m$$

In the next Sections, we will deal with some tensors in $\mathbb{R}^{K_1\times mK_2\times K_1\times mK_2}$. Let us denote this tensor $\mathcal{M}_m$, and suppose that $m=3$ (i.e., $\mathcal{M}_3 \in \mathbb{R}^{K_1\times 3K_2\times K_1\times 3K_2}$), by following the definitions above, the tensor $\mathcal{M}_3$ can be presented as follows

$$\mathcal{M}_3 = \Big|\,\big|\,|\mathcal{M}_{1,1} \quad \mathcal{M}_{1,2}|_2 \quad \mathcal{M}_{1,3}\big|_2,\,\big|\,|\mathcal{M}_{2,1} \quad \mathcal{M}_{2,2}|_2 \quad \mathcal{M}_{2,3}\big|_2,\,\big|\,|\mathcal{M}_{3,1} \quad \mathcal{M}_{3,2}\big|_2 \quad \mathcal{M}_{3,3}\big|_2\,\Big|_1,$$

where $\mathcal{M}_{i,j} \in \mathbb{R}^{K_1\times K_2\times K_1\times K_2}$ and the three 2-mode row block tensor given above have been constructed as mentioned in (3.6), where we concatenate every tensor $\mathcal{M}_{i,j}$ successively. To facilitate the notation, we will refer to $\mathcal{M}_m$ as

$$\mathcal{M}_3 = \left|\begin{array}{ccc} \mathcal{M}_{1,1} & \mathcal{M}_{1,2} & \mathcal{M}_{1,3} \\ \mathcal{M}_{2,1} & \mathcal{M}_{2,2} & \mathcal{M}_{2,3} \\ \mathcal{M}_{3,1} & \mathcal{M}_{3,2} & \mathcal{M}_{3,3} \end{array}\right| \in \mathbb{R}^{K_1\times 3K_2\times K_1\times 3K_2}$$

The tensors $\mathcal{M}_{i,j}$ could be defined from the tensor $\mathcal{M}_m$ by using the MATLAB colon as follows

$$\mathcal{M}_{i,j} = \mathcal{M}_3\left(:,(i-1)K_2+1:iK_2,\,:,(j-1)K_2+1:jK_2\right) \qquad \text{for } i,j = 1,2,3$$

This could be followed by a generalisation to $m \in \mathbb{N}$ (i.e., $\mathcal{M}_m \in \mathbb{R}^{K_1\times mK_2\times K_1\times mK_2}$) defined as follows

$$\mathcal{M}_m = \left|\begin{array}{cccc} \mathcal{M}_{1,1} & \mathcal{M}_{1,2} & \ldots & \mathcal{M}_{1,m} \\ \mathcal{M}_{2,1} & \mathcal{M}_{2,2} & \ldots & \mathcal{M}_{2,m} \\ \vdots & \ddots & \ddots & \vdots \\ \mathcal{M}_{m,1} & \mathcal{M}_{m,2} & \ldots & \mathcal{M}_{m,m} \end{array}\right| \in \mathbb{R}^{K_1\times mK_2\times K_1\times mK_2}$$

and in the same way, we can describe every tensor $\mathcal{M}_{i,j}$ using the MATLAB colon by the following

$\mathcal{M}_{i,j} = \mathcal{M}_m\left(:,(i-1)K_2+1:iK_2,\,:,(j-1)K_2+1:jK_2\right) \in \mathbb{R}^{K_1\times K_2\times K_1\times K_2}$ for $i,j = 1,2,\ldots,m$.

3.2 Multi-linear dynamical systems via Einstein Product

Consider the following discrete time MLTI system

$$\begin{cases} \mathcal{X}_{k+1} = \mathcal{A} *_N \mathcal{X}_k + \mathcal{B} *_M \mathcal{U}_k, \quad \mathcal{X}_0 = 0 \\ \mathcal{Y}_k = \mathcal{C} *_N \mathcal{X}_k \end{cases} \tag{3.8}$$

in the continuous case, a continuous MLTI system could be described as follows

$$\begin{cases} \dot{\mathcal{X}}(t) = \mathcal{A} *_N \mathcal{X}(t) + \mathcal{B} *_M \mathcal{U}(t), \quad \mathcal{X}_0 = 0 \\ \mathcal{Y}(t) = \mathcal{C} *_N \mathcal{X}(t) \end{cases} \tag{3.9}$$

where $\dot{\mathcal{X}}(t)$ is the derivative of the tensor $\mathcal{X}(t) \in \mathbb{R}^{J_1 \times \ldots \times J_N}$, $\mathcal{A} \in \mathbb{R}^{J_1 \times \ldots \times J_N \times J_1 \times \ldots \times J_N}$ is a square tensor, $\mathcal{B} \in \mathbb{R}^{I_1 \times \ldots \times J_N \times K_1 \times \ldots \times K_M}$ and $\mathcal{C} \in \mathbb{R}^{I_1 \times \ldots \times I_M \times J_1 \times \ldots \times J_N}$. The tensors $\mathcal{U}_k \in \mathbb{R}^{K_1 \times \ldots \times K_M}$ ($\mathcal{U}(t)$ in the continuous case) and $\mathcal{Y}_k \in \mathbb{R}^{I_1 \times \ldots \times I_M}$ ($\mathcal{Y}(t)$ in the continuous case) are the control and output tensors, respectively.

lemma 3.2.1. *Let $\Psi : \mathbb{T}_{j_1 \ldots j_N k_1 \ldots k_M} \to \mathbb{M}_{|J|,|K|}$ be the map defined in (3.1.7). Then the following properties hold :*

1. *The transformation Ψ is a bijection. Moreover, there exists a bijective inverse transformation $\Psi^{-1} : \mathbb{M}_{I_1 I_2, I_1 I_2}(\mathbb{R}) \to \mathbb{T}_{I_1, I_2, I_1, I_2}(\mathbb{R})$.*
2. *The map satisfies $\Psi(\mathcal{A} *_2 \mathcal{B}) = \Psi(\mathcal{A}) \cdot \Psi(\mathcal{B})$, where $\cdot$ refers to the usual matrix multiplication.*

The proof of this lemma can be found in [2].

Proposition 3.2.1. *The transfer function associated with dynamical system (3.8) is given by*

$$\mathcal{F}(s) = \mathcal{C} *_N (s\mathcal{I} - \mathcal{A})^{-1} *_N \mathcal{B} \tag{3.10}$$

Démonstration. Given the transformation Ψ defined in Definition 3.1.7, we can show that the multilinear system (3.8) could be transformed into an equivalent matrix linear dynamical system, defined by

$$\begin{cases} x_{k+1} = Ax_k + Bu_k, \quad x_0 = 0 \\ y_k = Cx_k \end{cases}$$

where A, B and C are the matrices resulting from the matricization of the tensors $\mathcal{A}, \mathcal{B}$ and $\mathcal{C}$ respectively by using Ψ. The same remark goes for the vectors x_k, u_k and y_k. System of the form (), can be represented in a frequency domain via Z-transform by the following transfer function.

$$F(s) = C(sI - A)^{-1}B$$

From the lemma, it is proved that for two matrices X and Y with appropriate dimensions, we have $\Psi^{-1}(XY) = \mathcal{X} * \mathcal{Y}$. Then, we get the following result

$$\begin{aligned}\mathcal{F}(s) = \Psi^{-1}(F(s)) &= \Psi^{-1}\left(C(sI - A)^{-1}B\right) \\ &= \Psi^{-1}(C) *_N \Psi^{-1}\left((sI - A)^{-1}\right) *_N \Psi^{-1}(B) \\ &= \mathcal{C} *_N (s\mathcal{I} - \mathcal{A})^{-1} *_N \mathcal{B}\end{aligned}$$

□

It's important to note that the theoretical concepts for the linear dynamical system, like stability, reachability, and observability, have been applied to MLTI systems. The results shown here are the same as those for matrices if we use the isomorphism Ψ mentioned earlier. If we start with a non-zero initial tensor state $\mathcal{X}_0$, similar to the matrix case, the solution to the discrete MLTI systems is given by :

$$\mathcal{X}_k = \mathcal{A}^{\star k} *_N \mathcal{X}_0 + \sum_{j=0}^{k-1} \mathcal{A}^{\star k-j-1} *_N \mathcal{B} *_M \mathcal{U}_j \tag{3.11}$$

where $\mathcal{A}^{\star k} = \underbrace{\mathcal{A} * \ldots * \mathcal{A}}_{k}$.

For example, the concept of stability in an LTI system can be extended to the MLTI case as follows :

Definition 3.2.1.

Given the MLTI system (). The system is called

1. *Asymptotically stable if* $\|\mathcal{X}_k\| \leq \kappa \|\mathcal{X}_0\|$ *for some* $\kappa > 0$.
2. *Stable if* $\|\mathcal{X}_k\| \to 0$ *as* $t \to \infty$.

Just like with LTI systems, to create a reduced model using Krylov subspace techniques, we focus on approximating the associated transfer function. This

approach can also be applied to MLTI systems. Here, we introduce a tensor global Arnoldi Krylov method using the Einstein product to construct this approximation, resulting in a reduced MLTI model. We use the classic Krylov subspace and also introduce the tensor extended Krylov subspace.

3.3 Tensor based global Krylov subspace methods

3.3.1 Classic Krylov subspace processing

The first step in the approximation of the associated transfer function () to the MLTI system () is that we rewrite it as

$$\mathcal{F}(s) = \mathcal{C} *_N \mathcal{X}$$

where $\mathcal{X}$ verify the following multi-linear system

$$(s\mathcal{I} - \mathcal{A}) *_N \mathcal{X} = \mathcal{B} \tag{3.12}$$

In order to find an approximation to $\mathcal{F}(s)$, it remains to find an approximation to the above multi-linear system, which can be done by using a projection Krylov subspace technique. Let us first define the m-th tensor Krylov subspace

$$\mathcal{K}_m(\mathcal{A}, \mathcal{B}) = \text{span}\left\{\mathcal{B}, \mathcal{A} * \mathcal{B}, \ldots, \mathcal{A}^{\star m-1} * \mathcal{B}\right\} \subseteq \mathbb{R}^{J_1 \times \ldots \times J_N \times K_1 \times \ldots \times K_M} \tag{3.13}$$

where $\mathcal{A}^{\star k} = \underbrace{\mathcal{A} * \ldots * \mathcal{A}}_{k}$. We focus on the case where $N = M = 2$ for simplicity. This means we will consider 4th-order tensors in the following sections. The results can be easily extended to cases where $N \geq 2$ and $M \geq 2$. From now on, we will call the Einstein product between two 4th-order tensors ($*_2$) simply "*".

The tensor global Arnoldi process is an analogue to the one described for matrices.

Algorithm 7 Tensor Global Arnoldi Algorithm

Input : $\mathcal{A} \in \mathbb{R}^{J_1 \times J_2 \times J_1 \times J_2}, \mathcal{B} \in \mathbb{R}^{J_1 \times J_2 \times K_1 \times K_2}$, and a fixed integer m.
Output : $\widetilde{\mathcal{V}}_{m+1} \in \mathbb{R}^{J_1 \times J_2 \times K_1 \times (m+1)K_2}$ and $\bar{H}_m \in \mathbb{R}^{m+1 \times m}$.
Set $\beta = \|\mathcal{B}\|_F$ and $\mathcal{V}_1 = \mathcal{B}/\beta$. (MATLAB notation $\widetilde{\mathcal{V}}_{m+1}(:,:,:,1:K_2) = \mathcal{V}_1$)
for $j = 1 \dots m$ **do**
    $\mathcal{W} = \mathcal{A} * \mathcal{V}_j$
    for $i = 1 \dots j$ **do**
        $h_{i,j} = \langle \mathcal{V}_i, \mathcal{W} \rangle$
        $\mathcal{W} = \mathcal{W} - h_{i,j}\mathcal{V}_i$
    end for
    $h_{j+1,j} = \|\mathcal{W}\|_F$
    if $h_{j+1,j} = 0$ **then**
        stop
    else
        $\mathcal{V}_{j+1} = \mathcal{W}/h_{j+1,j}$
    end if
    (MATLAB notation $\widetilde{\mathcal{V}}_{m+1}(:,:,:,jK_2+1:(j+1)K_2) = \mathcal{V}_{j+1}$)
end for

Definition 3.3.1. ***Product*** $\circledast$

Given a matrix $P \in \mathbb{R}^{m \times n}$ and a tensor $\mathcal{J} \in \mathbb{R}^{K_1 \times K_2 \times K_1 \times K_2}$, the resulted tensor $\mathcal{R} = P \circledast \mathcal{J}$ is defined as follows

1. $\mathcal{R} \in \mathbb{R}^{K_1 \times mK_2 \times K_1 \times nK_2}$.
2. $\mathcal{R}(i,:,i,:) = P \otimes \mathcal{J}(i,:,i,:)$*, for* $i = 1, \dots, K_1$*, where* $\otimes$ *is the Kronecker product.*

What we end up with, after m steps, is the following decomposition

$$\mathcal{A} * \widetilde{\mathcal{V}}_m = \widetilde{\mathcal{V}}_{m+1} * \left(\bar{H}_m \circledast \mathcal{I}_K\right) \tag{3.14}$$

where $\widetilde{\mathcal{V}}_{m+1} \in \mathbb{R}^{J_1 \times J_2 \times K_1 \times (m+1)K_2}$ contains $\mathcal{V}_i \in \mathbb{R}^{J_1 \times J_2 \times K_1 \times K_2}$ for $i = 1, \dots, m+1$, as it is explained in the algorithm above. $\bar{H}_m \in \mathbb{R}^{m+1 \times m}$ is a Hessenberg matrix of the following form

$$\bar{H}_m = \begin{bmatrix} h_{1,1} & \cdots & \cdots & h_{1,m} \\ h_{2,1} & \ddots & & \vdots \\ 0 & h_{3,2} & \ddots & \vdots \\ 0 & \cdots & \ddots & h_{m,m} \\ \vdots & & \ddots & 0 \\ 0 & \cdots & \cdots & h_{m+1,m} \end{bmatrix}$$

$\mathcal{I}_{\mathrm{K}} \in \mathbb{R}^{K_1 \times K_2 \times K_1 \times K_2}$ is the identity tensor, a diagonal tensor with entries

$$\mathcal{I}_K(i,j,k,l) = \delta_{i,k}\delta_{j,ll}, \quad \text{where} \quad \delta_{p,q} = 1, \text{if } p = q, \delta_{p,q} = 0, \text{otherwise.}$$

Notice that if $\mathcal{J}$ is a matrix in $\mathbb{R}^{K_1 \times K_2}$ then the product $\circledast$ becomes the Kronecker product $\otimes$.
The following result is used in the computational process described in the next sections. It is summarized in the following straightforward proposition.

Proposition 3.3.1. *Let $\widetilde{\mathcal{V}}_m \in \mathbb{R}^{J_1 \times J_2 \times K_1 \times mK_2}$ be the tensor generated from Algorithm . Let P, Q^T two matrices in $\mathbb{R}^{m \times n}, \mathcal{I}_K \in \mathbb{R}^{K_1 \times K_2 \times K_1 \times K_2}$ the identity tensor. Then we have*

$$\widetilde{\mathcal{V}}_m * (PQ \circledast \mathcal{I}_K) = \widetilde{\mathcal{V}}_m * ((P \circledast \mathcal{I}_K) * (Q \circledast \mathcal{I}_K)) \tag{3.15}$$

Notice that the computed $\mathcal{V}_i$'s from Algorithm 7 form an orthonormal basis of the tensor Krylov subspace (3.13); i.e. $\widetilde{\mathcal{V}}_m^T \diamond \widetilde{\mathcal{V}}_m = I_m$, where $I_m \in \mathbb{R}^{m \times m}$ is the identity matrix. The product $\diamond$ is an analogous to the one defined in the global Arnoldi process described for the matrix case in as follows

$$\tilde{\mathcal{V}}_m^T \diamond \widetilde{\mathcal{V}}_m = \begin{bmatrix} < \mathcal{V}_1, \mathcal{V}_1 > & < \mathcal{V}_1, \mathcal{V}_2 > & \ldots & < \mathcal{V}_1, \mathcal{V}_m > \\ < \mathcal{V}_2, \mathcal{V}_1 > & < \mathcal{V}_2, \mathcal{V}_2 > & \ldots & < \mathcal{V}_2, \mathcal{V}_m > \\ \vdots & \ddots & \ddots & \ldots \\ < \mathcal{V}_m, \mathcal{V}_1 > & \ldots & \ldots & < \mathcal{V}_m, \mathcal{V}_m > \end{bmatrix} \in \mathbb{R}^{m \times m}$$

where $\langle \cdot, \cdot \rangle$ is the inner-product defined in the previous section for tensors.

Proposition 3.3.2. *Given tensors* $\mathcal{E}, \mathcal{F}, \mathcal{G} \in \mathbb{R}^{J_1 \times J_2 \times K_1 \times mK_2}$ *and matrix* $S \in \mathbb{R}^{m \times m}$*. Then we have*

1. $(\mathcal{E} + \mathcal{F})^T \diamond \mathcal{G} = \mathcal{E}^T \diamond \mathcal{G} + \mathcal{F}^T \diamond \mathcal{G}$.
2. $\mathcal{E}^T \diamond (\mathcal{F} + \mathcal{G}) = \mathcal{E}^T \diamond \mathcal{F} + \mathcal{E}^T \diamond \mathcal{G}$.
3. $\mathcal{E}^T \diamond (\mathcal{F} * (S \circledast I_K)) = \left(\mathcal{E}^T \diamond \mathcal{F}\right) S$.

Based on these results, and using the definition of the product $\circledast$, we can easily prove the following proposition that will be of a great use in terms of simplification as we will show in the following sections.

Proposition 3.3.3. *Let* $\widetilde{\mathcal{V}}_{m+1}$ *be the orthonormal tensor given by Algorithm 7, then as for the matrix case, we have the classical algebraic relations*

$$\widetilde{\mathcal{V}}_{m+1}^T \diamond \left(\mathcal{A} * \widetilde{\mathcal{V}}_m\right) = \bar{H}_m, \quad \textit{and} \quad \widetilde{\mathcal{V}}_m^T \diamond \left(\mathcal{A} * \widetilde{\mathcal{V}}_m\right) = H_m \tag{3.16}$$

where $H_m \in \mathbb{R}^{m \times m}$ *is also a Hessenberg matrix extracted from* $\bar{H}_m$ *by deleting the last row.*

3.3.2 Extended Krylov subspace process

In the context of matrices, extended block and global Arnoldi methods tend to be more effective than the traditional methods. This is particularly true in model order reduction techniques. In this section, we will explain the extended tensor global Arnoldi process. But first, let's provide some definitions and comments about the extended tensor Krylov subspace. Let $\mathcal{A}$ and $\mathcal{V}$ be two tensors of suitable dimensions. The extended tensor global Krylov subspace of dimension $2m$, denoted by $\mathcal{K}_m^e(\mathcal{A}, \mathcal{V})$, is defined as follows :

$$\mathcal{K}_m^e(\mathcal{A}, \mathcal{V}) = \mathcal{K}_m(\mathcal{A}, \mathcal{V}) + \mathcal{K}_m\left(\mathcal{A}^{-1}, \mathcal{A}^{-1} * \mathcal{V}\right) \tag{3.17}$$

$$= \operatorname{span}\left\{\mathcal{A}^{-\star m} * \mathcal{V}, \ldots, \mathcal{A}^{-1} * \mathcal{V}, \mathcal{V}, \ldots, \mathcal{A}^{\star m-1} * \mathcal{V}\right\} \tag{3.18}$$

where $\mathcal{A}^{\star k} = \underbrace{\mathcal{A} * \ldots * \mathcal{A}}_{k}, \mathcal{A}^{\star -k} = \underbrace{\mathcal{A}^{-1} * \ldots * \mathcal{A}^{-1}}_{k}$ and $\mathcal{K}_m(\mathcal{A}, \mathcal{V})$ is the tensor classic global Krylov subspace defined in () associated with the pair $(\mathcal{A}, \mathcal{V})$.
It is important to note that if we use the previously defined isomorphism Ψ,

we obtain the matrices $\Psi(\mathcal{A}) = A \in \mathbb{R}^{J_1 J_2 \times J_1 J_2}$ and $\Psi(\mathcal{V}) = V \in \mathbb{R}^{J_1 J_2 \times K_1 K_2}$. This allows us to apply all the results from the matrix case to the tensor structure using the Einstein product. Now we will describe the process for constructing the tensor $\widetilde{\mathcal{V}}_{2m}$ associated with the extended global Krylov subspace defined earlier. This process is carried out using the extended Arnoldi algorithm.

Remark 3.3.1. *While our focus is specifically on 4th-order tensors, the following results also apply to higher-order tensors.*

Algorithm 8 Tensor Extended Global Arnoldi (TEGA) Algorithm

1: **Input :** $\mathcal{A} \in \mathbb{R}^{J_1 \times J_2 \times J_1 \times J_2}, \mathcal{B} \in \mathbb{R}^{J_1 \times J_2 \times K_1 \times K_2}$, and a fixed integer m.
2: **Output :** $\widetilde{\mathcal{V}}_{2(m+1)} \in \mathbb{R}^{J_1 \times J_2 \times K_1 \times 2(m+1)K_2}$ and $\bar{H}_m \in \mathbb{R}^{2(m+1)\times 2m}$.
3: Compute the global QR-decomposition of $|\mathcal{B}, \mathcal{A}^{-1} * \mathcal{B}|_2 \in \mathbb{R}^{J_1 \times J_2 \times K_1 \times 2K_2}$, i.e., $\mathrm{gQR}\left(|\mathcal{B}, \mathcal{A}^{-1} * \mathcal{B}|_2\right) = [\mathcal{V}_1, \omega]$. (MATLAB notation $\widetilde{\mathcal{V}}_{2(m+1)}(:,:,:,1:2K_2) = \mathcal{V}_1$, $\omega = \begin{bmatrix} \omega_{1,1} & \omega_{1,2} \\ 0 & \omega_{2,2} \end{bmatrix} \in \mathbb{R}^{2\times 2}$)
4: **for** $j = 1 \cdots m$ **do**
5: Set $\mathcal{W} = \left|\mathcal{A} * \mathcal{V}_j^1, \mathcal{A}^{-1} * \mathcal{V}_j^2\right|_2$, where
6: $\mathcal{V}_j^1 = \mathcal{V}_j(:,:,:,(j-1)K_2 + 1 : jK_2)$
7: $\mathcal{V}_j^2 = \mathcal{V}_j(:,:,:,jK_2 + 1 : (j+1)K_2)$
8: **for** $i = 1, \ldots, j$ **do**
9: $H_{i,j} = \mathcal{V}_i^T \diamond \mathcal{W} \in \mathbb{R}^{2\times 2}$
10: $\mathcal{W} = \mathcal{W} - \mathcal{V}_i * (H_{i,j} \circledast \mathcal{I}_K)$
11: **end for**
12: Compute the global $\mathrm{QR}(\mathrm{gQR})$ of $\mathcal{W}$, i.e., $\mathrm{gQR}(\mathcal{W}) = [\mathcal{V}_{j+1}, H_{j+1,j}]$
13: **end for**

Algorithm creates an orthonormal tensor basis called $\widetilde{\mathcal{V}}_{2(m+1)}$. This basis is associated with the extended global Krylov subspace for the pair $(\mathcal{A}, \mathcal{B})$. It ensures that for $i, j = 1, \ldots, m$, the following conditions are met :

$$\mathcal{V}_i^T \diamond \mathcal{V}_j = 0_{2\times 2} \quad (i \neq j) \quad \text{and} \quad \mathcal{V}_i^T \diamond \mathcal{V}_i = I_{2\times 2}$$

An upper Hessenber matrix $\bar{H}_m \in \mathbb{R}^{2(m+1)\times 2m}$ is also generated with upper block matrices $H_{i,j} \in \mathbb{R}^{2\times 2}$.

• In Step 3 and Step 10 of Algorithm 8, we compute a global QR decomposition to get the element $\mathcal{V}_j$ of the tensor basis $\widetilde{\mathcal{V}}_{2(m+1)}$. We use this decomposition to get $|\mathcal{B}, \mathcal{A}^{-1} * \mathcal{B}|_2 = \mathcal{V}_1 * (\omega \circledast \mathcal{I}_K)$ in Step 3 and $\mathcal{W} = \mathcal{V}_{j+1} * (H_{j+1,j} \circledast \mathcal{I}_K)$ in Step 10, where product $\circledast$ is the one described in

• From steps 5 to 9 , we obtain the following relations

$$\begin{aligned}\mathcal{V}_{j+1} * (H_{j+1,j} \circledast I_K) &= \left|\mathcal{A} * \mathcal{V}_j^1, \mathcal{A}^{-1} * \mathcal{V}_j^2\right|_2 - \sum_{i=1}^{j} \mathcal{V}_i * (T_{i,j} \circledast \mathcal{I}_K) \\ H_{j+1,j} &= \mathcal{V}_{j+1}^T \diamond \left|\mathcal{A} * \mathcal{V}_j^1, \mathcal{A}^{-1} * \mathcal{V}_j^2\right|_2 \\ &= \mathcal{V}_{j+1}^T \diamond \sum_{i=1}^{j+1} \mathcal{V}_i * (H_{i,j} \circledast \mathcal{I}_K)\,.\end{aligned}$$

After m steps of Algorithm we can show that

$$\begin{aligned}\mathcal{A} * \widetilde{\mathcal{V}}_{2m} &= \widetilde{\mathcal{V}}_{2(m+1)} * \left(\bar{T}_m E_m^T \circledast \mathcal{I}_K\right) \\ &= \widetilde{\mathcal{V}}_{2m} * (T_m \circledast \mathcal{I}_k) + \mathcal{V}_{m+1} * \left(T_{m+1,m} E_m^T \circledast \mathcal{I}_K\right)\end{aligned}$$

where E_m is the last $2m \times 2$ block column of the identity matrix $I_m \in \mathbb{R}^{2m\times 2m}$, $\bar{T}_m = \tilde{\mathcal{V}}_{2(m+1)}^T \diamond \left(\mathcal{A} * \widetilde{\mathcal{V}}_{2m}\right) \in \mathbb{R}^{2(m+1)\times 2m}$ and T_m is a $2m \times 2m$ block Hessenberg matrix defined by $T_m = \widetilde{\mathcal{V}}_{2m}^T \diamond \left(\mathcal{A} * \widetilde{\mathcal{V}}_{2m}\right) \in \mathbb{R}^{2m\times 2m}$, where $T_{i,j} = \mathcal{V}_i^T \diamond (\mathcal{A} * \mathcal{V}_j)$, for $i, j = 1, \ldots, m$. We summarize in the following result a recursion to compute T_m by avoiding additional tensor products with $\mathcal{A}$.

Proposition 3.3.4. *Let $\bar{H}_m = [h_{:,1}, \ldots, h_{:,m}] \in \mathbb{R}^{2(m+1)\times 2m}$ be the upper Hessenberg matrix generated from Algorithm 13 where $h_{:,i} \in \mathbb{R}^{2(m+1)\times 1}$ is the ith-column. A simple computation of the upper Hessenberg matrix $\bar{T}_m = [t_{:,1}, \ldots, t_{:;m}] \in \mathbb{R}^{2(m+1)\times 2m}$ is as follows.*

The odd columns of the $\bar{T}_m$ are such that

$$t_{:,2j-1} = h_{:,2j-1}, \quad \textit{for } j = 1, \ldots, m$$

and the even column are described as

$$\begin{aligned} t_{:,2} &= \frac{1}{\omega_{22}} \left(\omega_{11} e_1^{2(m+1)} - \omega_{12} h_{:,1}\right) \\ t_{:,2j+2} &= \frac{1}{h_{2j+2,2j}} \left(e_{2j}^{2(m+1)} - t_{:,1:2j+1} h_{1:2j+1,2j}\right) \quad \textit{for } j = 1, \ldots, m-1 \end{aligned} \tag{3.19}$$

where $e_i^{(k)}$ *is the ith column vector of the identity matrix* I_k *and* $\omega_{1,1}, \omega_{1,2}$ *and* $\omega_{2,2}$ *are as defined in step 2 from Algorithm 7.*

3.3.3 Order reduction via projection

Based on the tensor classic and extended Krylov subspace processes described above, we are now able to provide a reduction process to get a reduced MLTI system to the original one (3.8). We recall our original MLTI system

$$\begin{aligned} \mathcal{X}_{k+1} &= \mathcal{A} * \mathcal{X}_k + \mathcal{B} * \mathcal{U}_k, \quad \mathcal{X}_0 = 0 \\ \mathcal{Y}_k &= \mathcal{C} * \mathcal{X}_k \end{aligned}$$

where the tensors $\mathcal{A} \in \mathbb{R}^{J_1 \times J_2 \times J_1 \times J_2}$ and $\mathcal{B}, \mathcal{C}^T \in \mathbb{R}^{J_1 \times J_2 \times K_1 \times K_2}$. For simplicity, we consider product $*$ as the Einstein product $*_2$ between 4 th-order tensors. The associated transfer function mentioned previously in Proposition 3.2.1 is given as

$$\mathcal{F}(s) = \mathcal{C} * (s\mathcal{I} - \mathcal{A})^{-1} * \mathcal{B}$$

We look to construct the following reduced MLTI system

$$\begin{aligned} \hat{\mathcal{X}}_{k+1} &= \hat{\mathcal{A}} * \hat{\mathcal{X}}_k + \hat{\mathcal{B}} * \hat{\mathcal{U}}_k, \quad \mathcal{X}_0 = 0 \\ \hat{\mathcal{Y}}_k &= \hat{\mathcal{C}} * \hat{\mathcal{X}}_k \end{aligned}$$

by using Proposition 3.2.1, we can describe the associated transfer function as follows

$$\hat{\mathcal{F}}(s) = \hat{\mathcal{C}} * (s\hat{\mathcal{I}} - \hat{\mathcal{A}})^{-1} * \hat{\mathcal{B}}$$

To check how accurate the reduced system is, we can calculate the error-norm $\|\mathcal{F} - \hat{\mathcal{F}}\|$ using a specific norm. Just like the process used for matrices, we can do something similar with tensors.

First, we look at one of the Krylov subspaces mentioned earlier, which are

used to project the initial MLTI system. These subspaces have dimensions m and $2m$. We use the tensor $\mathcal{V}_m$ as the basis for one of these subspaces.

By approximating the full-order state $\mathcal{X}_k$ as $\mathcal{V}_m * \hat{\mathcal{X}}_k$, and using a method similar to the one used for matrices called the Petrov-Galerkin condition technique, we can simplify the system as follows :

$$\begin{cases} \mathcal{V}_m^T * \left(\mathcal{V}_m * \hat{\mathcal{X}}_{k+1} - \mathcal{A} * \mathcal{V}_m * \hat{\mathcal{X}}_k - \mathcal{B} * \mathcal{U}_k \right) = 0 \\ \mathcal{Y}_k = \mathcal{C} * \mathcal{V}_m * \hat{\mathcal{X}}_k \end{cases}$$

finally, we obtain the desired reduced MLTI system, with the following tensorial structure

$$\hat{\mathcal{A}} = \mathcal{V}_m^T * (\mathcal{A} * \mathcal{V}_m), \quad \hat{\mathcal{B}} = \mathcal{V}_m^T * \mathcal{B}, \quad \hat{\mathcal{C}} = \mathcal{C} * \mathcal{V}_m \tag{3.20}$$

We will now clearly explain the tensor structure given in equation (3.20) using the two processes based on the Krylov subspaces we discussed earlier. Our goal is to find an approximation $\hat{\mathcal{F}}$ of $\mathcal{F}$, which means we need to find an approximation $\mathcal{X}_m$ of $\mathcal{X}$ that satisfies the multi-linear system (3.12). This approximation should belong to either the tensor classic Krylov subspace, $\mathcal{K}_m(\mathcal{A}, \mathcal{B})$, or the tensor extended Krylov subspace, $\mathcal{K}_m^e(\mathcal{A}, \mathcal{B})$. Here is how we find the approximation using the tensor classic Krylov subspace :

$$\mathcal{X}_m = \widetilde{\mathcal{V}}_m * (y_m \circledast \mathcal{I}_K) \in \mathbb{R}^{J_1 \times J_2 \times K_1 \times K_2}, \tag{3.21}$$

$$\mathcal{R}_m = \mathcal{B} - (s\mathcal{I} - \mathcal{A}) * \mathcal{X}_m \perp \mathcal{K}_m(\mathcal{A}, \mathcal{B}) \tag{3.22}$$

where y_m is a vector with m-components and $\widetilde{\mathcal{V}}_m$ is the orthonormal basis associated with the tensor Krylov subspace (3.13). We know from the decomposition (3.14) that

$$\mathcal{A} * \widetilde{\mathcal{V}}_m = \widetilde{\mathcal{V}}_m * (H_m \circledast \mathcal{I}_K) + h_{m+1,m} \mathcal{V}_{m+1} * \left(e_m^T \circledast \mathcal{I}_K \right)$$

and by using the fact $\mathcal{B} = \|\mathcal{B}\| \mathcal{V}_1$ and Proposition 3.3.2, the equality (3.22) could be simplified as follows

$$\widetilde{\mathcal{V}}_m^T \diamond \left(\mathcal{B} - (s\mathcal{I} - \mathcal{A}) * \left(\widetilde{\mathcal{V}}_m * (y_m \circledast \mathcal{I}_K)\right)\right. = 0,$$

$$\widetilde{\mathcal{V}}_m^T \diamond \mathcal{B} = \widetilde{\mathcal{V}}_m^T \diamond \left(\left(s\widetilde{\mathcal{V}}_m - \mathcal{A} * \widetilde{\mathcal{V}}_m\right) * (y_m \circledast \mathcal{I}_K)\right),$$

$$\|\mathcal{B}\| e_1^m = (sI_m - H_m)\, y_m$$

$$y_m = (sI_m - H_m)^{-1} \|\mathcal{B}\| e_1^m.$$

Now, we can have an explicit expression of the approximation of $\mathcal{F}(s)$ denoted by $\mathcal{F}_m(s)$ and described as

$$\begin{aligned}\mathcal{F}(s) \approx \hat{\mathcal{F}}(s) = \mathcal{C} * \mathcal{X}_m &= \mathcal{C} * \left(\widetilde{\mathcal{V}}_m * (y_m \circledast \mathcal{I}_K)\right) \\ &= \mathcal{C} * \left(\widetilde{\mathcal{V}}_m * \left(\left((sI_m - H_m)^{-1} \|\mathcal{B}\| e_1^m\right) \circledast \mathcal{I}_K\right)\right) \\ &= \mathcal{C} * \widetilde{\mathcal{V}}_m * (s\mathcal{I}_K^m - (H_m \circledast \mathcal{I}_K))^{-1} * (\|\mathcal{B}\| e_1^m \circledast \mathcal{I}_K)\end{aligned}$$

where $\mathcal{I}_K^m \in \mathbb{R}^{K_1 \times mK_2 \times K_1 \times mK_2}$ is the identity tensor. Now, from the approximation $\hat{\mathcal{F}}(s)$, we can conclude the reduced MLTI system

$$\begin{aligned}\hat{\mathcal{X}}_{k+1} &= \hat{\mathcal{A}} * \hat{\mathcal{X}}_k + \hat{\mathcal{B}} * \mathcal{U}_k \\ \hat{\mathcal{Y}}_k &= \hat{\mathcal{C}} * \hat{\mathcal{X}}_k\end{aligned}$$

with the associated tensor structure

$$\hat{\mathcal{A}} = H_m \circledast \mathcal{I}_K, \quad \hat{\mathcal{B}} = \|\mathcal{B}\| e_1^m \circledast \mathcal{I}_K, \quad \hat{\mathcal{C}} = \mathcal{C} * \widetilde{\mathcal{V}}_m \tag{3.23}$$

In order to quantify how well the approximated transfer function $\hat{\mathcal{F}}(s)$ is closer to the original one $\mathcal{F}(s)$, we propose in the following a simplified expression to the error norm $\|\mathcal{F} - \hat{\mathcal{F}}\|$.

Proposition 3.3.5. *Let $\mathcal{F}(s)$ and $\hat{\mathcal{F}}(s)$ be the two transfer functions associated with the original MLTI and the reduced one respectively. Based on the previous results, we get*

$$\|\mathcal{F}(s)-\hat{\mathcal{F}}(s)\| \leq \left\|\mathcal{C} * \widetilde{\mathcal{V}}_m * (s\mathcal{I}-\mathcal{A})^{-1}\right\| * \left\|h_{m+1,m}\left(\left(e_m^T\left(sI_m-H_m\right)^{-1}\right) \circledast \mathcal{I}_K\right) * \mathcal{B}_m\right\|$$

Proof 3.3.1. *The error $\mathcal{F}(s)-\hat{\mathcal{F}}(s)$ could be expressed as*

$$\begin{aligned}\mathcal{F}(s)-\hat{\mathcal{F}}(s) &= \mathcal{C} * (s\mathcal{I}-\mathcal{A})^{-1} * \mathcal{B} - \mathcal{C}_m * (s\mathcal{I}_{\mathrm{K}}^m - \mathcal{A}_m)^{-1} * \mathcal{B}_m \\ &= \mathcal{C} * (s\mathcal{I}-\mathcal{A})^{-1} * \left[\mathcal{B} - (s\mathcal{I}-\mathcal{A}) * \widetilde{\mathcal{V}}_m * (s\mathcal{I}_K^m - \mathcal{A}_m)^{-1} * \mathcal{B}_m\right]\end{aligned}$$

by using the decomposition (3.14), we obtain

$$\begin{aligned}\mathcal{F}(s)-\hat{\mathcal{F}}(s) &= \mathcal{C} * (s\mathcal{I}-\mathcal{A})^{-1} * [\mathcal{B} - \underbrace{\widetilde{\mathcal{V}}_m * B_m}_{\mathcal{B}} + h_{m+1,m}\mathcal{V}_{m+1} * \left(e_m^T \circledast \mathcal{I}_K\right) * (s\mathcal{I}_K^m - \mathcal{A}_m)^{-1} * \mathcal{B}_m] \\ &= \mathcal{C} * (s\mathcal{I}-\mathcal{A})^{-1} * \left[h_{m+1,m}\mathcal{V}_{m+1} * \left(\left(e_m^T\left(sI_m-H_m\right)^{-1}\right) \circledast \mathcal{I}_K\right) * \mathcal{B}_m\right]\end{aligned}$$

this concludes the proof.

If we consider the tensor extended global Krylov subspace $\mathcal{K}_m^e(\mathcal{A},\mathcal{B})$ as described in equation (3.17), we can obtain the approximated transfer function $\hat{\mathcal{F}}$ by following the same process used for the tensor classic global Krylov subspace. The approximated transfer function $\hat{\mathcal{F}}$ will then have the following form :

$$\hat{\mathcal{F}}(s) = \hat{\mathcal{C}} * \left(s\mathcal{I}_{2m} - \hat{\mathcal{A}}\right)^{-1} * \hat{\mathcal{B}} \tag{3.24}$$

with the associated structure matrices

$$\hat{\mathcal{A}} = T_m \circledast \mathcal{I}_{\mathrm{K}}, \quad \hat{\mathcal{B}} = \left(\tilde{\mathcal{V}}_{2m}^T \diamond \mathcal{B}\right) \circledast \mathcal{I}_K, \quad \hat{\mathcal{C}} = \mathcal{C} * \tilde{\mathcal{V}}_{2m} \tag{3.25}$$

From Algorithm 8, we know that

— $|\mathcal{B}, \mathcal{A}^{-1} * \mathcal{B}|_2 = \mathcal{V}_1 * (\omega \circledast \mathcal{I}_K)$ where $\omega \in \mathbb{R}^{2\times 2}$ is an upper triangular matrix from Algorithm 8, then,

$$\mathcal{B}_m = \left(\omega_{1,1} e_1^{(2m)}\right) \circledast \mathcal{I}_K$$

where $e_1^{(2m)}$ is the first column of the identity matrix I_{2m}.

— $T_m = \widetilde{\mathcal{V}}_{2m}^T \diamond \left(\mathcal{A} * \widetilde{\mathcal{V}}_{2m}\right)$ a $2m \times 2m$ is a Hessenberg matrix that can be computed in a efficient way using Proposition 3.3.4 and $\widetilde{\mathcal{V}}_{2m}$ is the tensor computed by Algorithm 8.

Remark 3.3.2. *It is worth noting that in the case of a continuous-time MLTI system described as*

$$\begin{cases} \dot{\mathcal{X}}(t) = \mathcal{A} * \mathcal{X}(t) + \mathcal{B} * \mathcal{U}(t), \quad \mathcal{X}_0 = 0 \\ \mathcal{Y}(t) = \mathcal{C} * \mathcal{X}(t) \end{cases}$$

the process described above for constructing a discrete-time reduced MLTI system can be also applied to the continuous-time MLTI system, we end up by the following reduced system

$$\begin{cases} \dot{\mathcal{X}}_m(t) = \mathcal{A}_m * \mathcal{X}_m(t) + \mathcal{B}_m * \mathcal{U}_t \\ \mathcal{Y}_m(t) = \mathcal{C}_m * \mathcal{X}_m(t) \end{cases}$$

where the coefficient tensorial are as follows

$$\mathcal{A}_m = \mathcal{V}_m^T * (\mathcal{A} * \mathcal{V}_m), \quad \mathcal{B}_m = \mathcal{V}_m^T * \mathcal{B}, \quad \mathcal{C}_m = \mathcal{C} * \mathcal{V}_m$$

3.3.4 Numerical Test

In this section, we show some examples to test how well the algorithms work using tensor extended global Krylov subspaces. The results were obtained using Python on a computer with an Intel® Core i7 processor running at 2.3GHz and 16GB of RAM. Similar to the matrix case, we can define a norm called the H_∞-norm. This norm helps us measure and plot the two transfer functions to compare how close the reduced MLTI system is to the original one.

Similar to the matrix case, we can define the following norm called the h_∞-norm to describe the magnitudes needed for plotting the two transfer functions and to see how close the reduced MLTI system is to the original one

$$\|\mathcal{F}(.)\|_\infty = \|\Psi(\mathcal{F}(.))\|_\infty = \sup_{\theta \in [0, 2\pi]} \sigma_{\max}(\Psi(\mathcal{F}(e^{j\theta}))),$$

where $\sigma_{\max}(A)$ is the largest singular value of the matrix A.

Exemple 1.[10]

We use the following data for our example :

— $\mathcal{A} \in \mathbb{R}^{N\times N\times N\times N}$, with $N = 100$. Here, $\mathcal{A}$ is derived by tensorizing a triangular matrix $A \in \mathbb{R}^{N\times N\times N\times N}$ constructed using Python.

— $\mathcal{B}, \mathcal{C}^T \in \mathbb{R}^{N\times K_1\times K_2}$ are selected as sparse and random tensors respectively, where $K_1 = 3$ and $K_2 = 5$.

Choosing $m = 5$, which is the dimension of the extended global Krylov subspace for the tensor, allows us to project the original MLTI system effectively.

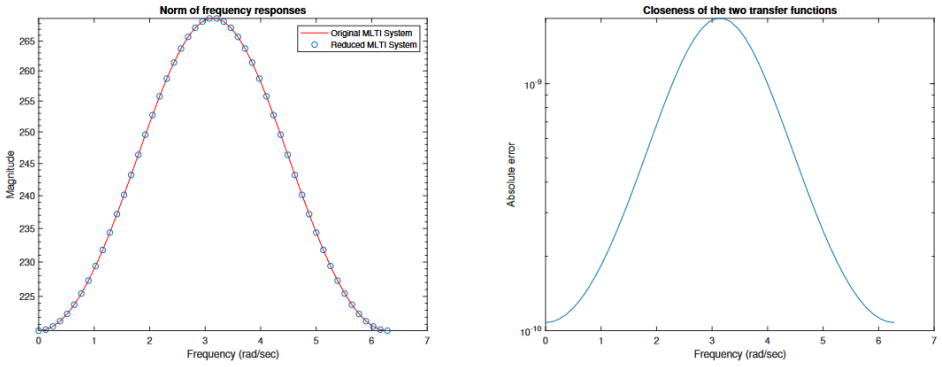

FIGURE 3.10 – The frequency responses of the original and reduced MLTI systems (left) and the error norms $\|F - F_m\|_\infty$ (right).

Exemple 2. [10]

This example concerns the evolution of heat distribution in a solid medium over time. The partial differential equation that describes this evolution is given by the following equation and named the 2D heat equation

$$\begin{cases} \frac{\partial}{\partial t}\phi(t,x) = c^2\frac{\partial^2}{\partial x^2}\phi(t,x) + \delta(x)u_t, & x \in D \text{ (square } D = [-\pi^2, \pi]), \\ \phi(t,x) = 0, & x \in \partial D, \end{cases} \tag{3.26}$$

where $c > 0$, u_t is a one-dimensional control input and $\delta(x)$ is the Dirac delta function centred at zeros.

A second-order central difference is used to approximate the Laplacian and a first-order difference in time, this leading to a MLTI system, where $X_k \in \mathbb{R}^{N\times N}$ called the 2D-temperature field at instance k. For further information, we

refer the readers to [7].

We consider the following system tensors

— The tensor $\mathcal{A} \in \mathbb{R}^{N\times N\times N\times N}$ with $N = 128$ is the tensorization of $\frac{c^2\Delta t}{h^2}\Delta_{dd} \in \mathbb{R}^{N^2\times N^2}$, where Δ_{dd} is the discrete Laplacian on a rectangular grid with a Dirichlet boundary condition.

— The constructed tensor $\mathcal{B}$ in [7] is of dimension $N \times N \times 1 \times 1$. Here, we change $\mathcal{B}$ and choose it as a sparse tensor in $\mathbb{R}^{N\times N\times K_1\times K_2}$ with $K_1 = 3$ and $K_2 = 5$.

— To complete the system, we will need the output tensor $\mathcal{C}$. We choose it as a random tensor in $\mathbb{R}^{K_1\times K_2\times N\times N}$ with $K_1 = 3$ and $K_2 = 5$.

— We set $m = 2$, the associated dimension of the extended tensor Krylov subspace.

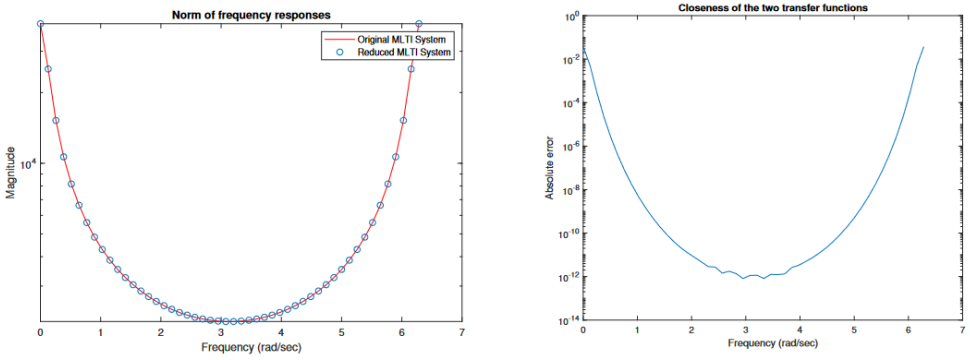

FIGURE 3.11 – The frequency responses of the original and reduced MLTI systems (left) and the error norms $\|\mathcal{F} - \mathcal{F}_m\|_\infty$ (right).

Chapitre 4

Conclusion

This report explored various methods for handling large-scale linear and multilinear dynamical systems. We focused on projection techniques, particularly those based on Krylov processes, for model reduction in large-scale Multi-Input Multi-Output (MIMO) linear time-invariant dynamical systems.

We began by introducing linear dynamical systems and their key properties. We then delved into four main approaches for model reduction :

1. The global Arnoldi method
2. The block Arnoldi method
3. The extended Global Arnoldi method
4. The extended Block Arnoldi method

For each method, we explained the algorithm, its application to MIMO systems, and how it can be used for model reduction. We also provided numerical tests to demonstrate the effectiveness of these methods.

Next, we expanded our focus to multi-linear dynamical systems. We introduced tensor mathematics, including various tensor operations and products. We then explored how these concepts can be applied to multi-linear dynamical systems, particularly using the Einstein product.

We developed tensor-based global Krylov subspace methods for both classic and extended Krylov subspaces. These methods allow us to reduce the order of multi-linear dynamical systems while preserving important system properties.

The numerical tests we conducted showed that these methods can effectively

reduce the order of large-scale systems while maintaining a good approximation of the original system's behavior.

Perspectives :

1. Trying to use the T-product in place of the Einstein product for multilinear dynamical systems. This could potentially lead to similar or even better results, and it would be interesting to compare the performance and computational efficiency of these two approaches.

2. Studying the preservation of specific system properties (like stability or passivity) in the reduced models.

3. Exploring the use of machine learning techniques in combination with these model reduction methods.

4. Applying these techniques to real-world problems in areas such as control systems, signal processing, or computational fluid dynamics.

Bibliographie

[1] Abidi, O. (2016, December). Méthodes de sous-espaces de Krylov rationnelles pour le contrôle et la réduction de modèles. Université du Littoral Côte d'Opale.

[2] Brazell, M., Li, N., Navasca, C., and Tamon, C. (2013). Solving multilinear systems via tensor inversion. SIAM Journal on Matrix Analysis and Applications, 34(2), 542-570.

[3] Antoulas, A. C. (2005). An overview of approximation methods for large-scale dynamical systems. Annual reviews in Control, 29(2), 181-190.

[4] Antoulas, A. C. (2005). Approximation of large-scale dynamical systems. Society for Industrial and Applied Mathematics.

[5] Barkouki, H. (2016). Rational Lanczos-type methods for model order reduction (Doctoral dissertation, Université du Littoral Côte d'Opale ; Université Cadi Ayyad (Marrakech, Maroc). Faculté des sciences et techniques Guéliz).

[6] Braman, K. (2010). Third-order tensors as linear operators on a space of matrices. Linear Algebra and its Applications, 433(7), 1241-1253.

[7] Chen, C., Surana, A., Bloch, A., and Rajapakse, I. (2019). Data-driven model reduction for multilinear control systems via tensor trains. arXiv preprint arXiv :1912.03569.

[8] El Ichi, A. (2021). Tensor Computation with Applications (Doctoral dissertation, Université du Littoral Côte d'Opale ; Université Mohammed V (Rabat). Faculté des sciences).

[9] Hamadi, M. A. (2023). Krylov-based subspaces methods for large-scale dynamical systems and data-driven model reduction (Doctoral dissertation, Université du Littoral Côte d'Opale ; Université Mohammed VI Polytechnique (Benguérir, Maroc)).

[10] Hamadi, M. A., Jbilou, K., and Ratnani, A. (2023). A model reduction method for large-scale linear multidimensional dynamical systems. arXiv preprint arXiv :2305.09361.

[11] He, Z. H., Ng, M. K., and Zeng, C. (2021). Generalized singular value decompositions for tensors and their applications. Numer. Math. Theory Methods Appl, 14(3), 692-713.

[12] Heyouni, M., and Jbilou, K. (2006). Matrix Krylov subspace methods for large scale model reduction problems. Applied Mathematics and Computation, 181(2), 1215-1228.

[13] Heyouni, M. (2010). Extended Arnoldi methods for large low-rank Sylvester matrix equations. Applied numerical mathematics, 60(11), 1171-1182.

[14] Jbilou, K. (2023). Computational methods for model reduction in large-scale MIMO dynamical systems.

[15] Kilmer, M. E., and Martin, C. D. (2011). Factorization strategies for third-order tensors. Linear Algebra and its Applications, 435(3), 641-658.

[16] Lu, C. (2018). Tensor-tensor product toolbox. arXiv preprint arXiv :1806.07247.

[17] Mehrmann, V., and Penzl, T. (1998). Benchmark collections in SLICOT. SLICOT Working Note, ESAT, Belgium.

[18] Penzl, T. (2000). A MATLAB Toolbox for Large Lyapunov and Riccati Equations, Model Reduction Problems, and Linear–Quadratic Optimal Control Problems. Software available at https ://www.tu-chemnitz.de/sfb393/lyapack.

[19] Saad, Y. (1989). Numerical solution of large Lyapunov equations (No. NAS 1.26 : 180359).