14 | 2003

Low-rank approximation and model reduction

Younès Chahlaoui

Décembre 2003

Thèse présentée en vue de l'obtention du grade
de docteur en sciences appliquées

Faculté des sciences
appliquées

Université catholique de Louvain

Younès Chahlaoui

Low-rank approximation and model reduction

UCL

Université Catholique De Louvain
Faculté des sciences appliquées
Département d'ingénierie mathématique

Center for Systems Engineering and Applied Mechanics

Low-Rank Approximation and Model Reduction

Younès Chahlaoui

December 2003

Université Catholique De Louvain
Faculté des sciences appliquées
Département d'ingénierie mathématique

Center for Systems Engineering and Applied Mechanics

Low-rank approximation and model reduction

Younès Chahlaoui

Thesis submitted in partial fulfillment
of the requirements for the degree of
Docteur en sciences appliquées

Dissertation committee:
Prof. Mohammed Elarbi Achhab
Prof. Patrick Dewilde
Prof. Michel Gevers (chair)
Prof. Dirk Roose
Prof. Paul Van Dooren (advisor)
Prof. Vincent Wertz

December 2003

Et dis :
“Ô mon Seigneur, accroîs mes connaissances !”

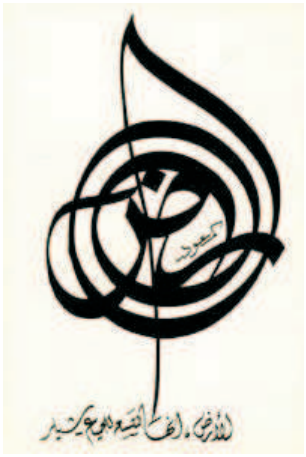
La quête du savoir est une démarche qui revêt un caractère religieusement obligatoire pour moi. En effet, elle fait partie des préceptes du Coran : ***“Dis : “sont-ils égaux ceux qui savent et ceux qui ne savent pas ?”; seuls les doués d’intelligence se rappellent”****. Il suffit pour souligner l’importance de l’acquisition du savoir dans la tradition islamique, de rappeler que le premier mot révélé du Coran est cette injonction :* ***“lis !”****. Les textes coraniques présentent le savoir comme étant inséparable aussi bien de la foi en Dieu, que d’une confiance de l’homme et en sa capacité maîtriser le monde. Etant lui-même la source et la finalité, l’être humain (homme et femme) est appelé à valoriser sa foi en Dieu par la quête des sciences, car le Coran décrit l’ignorance comme étant synonyme de la non-croyance en Dieu. Un hadith du Prophète rapporte :* ***“Recherche la science, même en Chine ”****. C’est dans ce cadre là que la quête du savoir et de la connaissance est un devoir religieux qui donne un sens à ma vie. D’autant plus que les personnes les plus chères à mon coeur me soutiennent depuis le début en me motivant à continuer et à aller toujours le plus loin possible dans tout ce que je fais.*

Younès Chahlaoui
Louvain-La-Neuve,
December 9, 2003

To my mother Naïma
my father Mohammed
Aïcha, Sawsanne, Tata, Yasmine, and Yassine
my angel
and all my family

dedicated to the memory of
my grandfather Abdelkader
Baassidi, Lala
Victor Vermeulen († 30/03/2003)



"Et toi mon coeur pourquoi bats-tu?"
Jean d'Ormesson

Acknowledgments

Completing this doctoral work has been a wonderful and often overwhelming experience. It is hard to know whether it has been grappling with the applied mathematics itself which has been the real learning experience, or grappling with how to write a paper, give a coherent talk, work in a group, help in teaching, code intelligibly, debug a program, stay up until the equations start to dance in front of my eyes, and... stay, um... focussed.

I have been very privileged to have undoubtedly the most intuitive, smart and supportive advisor anyone could ask for, namely Paul Van Dooren. Ever since I learned from him what model reduction, Gramians and many other things were, I have been stimulated and excited by his constant flow of good ideas. Paul has an ability to cut through reams of algebra with a single visual explanation that I will always admire, and I have learned a great deal of numerics from him. He has fostered certainly the most open, friendly, collaborative and least competitive research group in the model reduction world. He has also known when (and how) to give me a little push in the forward direction when I needed it.

I am indebted to all the members of the jury of this thesis who undoubtedly contributed to improve the quality of this manuscript. I thank Profs. Elarbi Achhab, Patrick Dewilde, Dirk Roose, Vincent Wertz and Michel Gevers for their attentive and critical reading of a previous version of this manuscript. Their advices were most valuable. I also thank Prof. Michel Gevers for accepting the role of Chair of the jury.

Throughout my four years, I was supported mainly within the framework of a collaboration agreement between CESAME (Université catholique de Louvain, Belgium) and LINMA of the Faculty of Sciences (Université Chouaib Doukkali, Morocco), funded by the Secretary of the State for Development Cooperation and by the CIUF (Conseil Interuniversitaire de la Communauté Française, Belgium), and coordinated by Vincent Wertz and Elarbi Achhab. I would like to thank both of them for giving me the chance to do this work. I also received support from the Belgian Programme on Inter-university Poles of Attraction (IAP Phases IV and V), coordinated by Prof. Michel Gevers, and initiated by the Belgian State, Prime Minister's Office for Science, Technology and Culture.

CESAME's PhD students and post-docs, both past and present, make a superb research group. The ability to bounce ideas off so many excellent minds has been priceless. My most intense collaboration has been with Antoine Vandendorpe and Damien Lemonnier, with whom discussions were always a real pleasure and have taught me a lot. Other people like Cédric Delattre, Agnès Provost, Sophie Beguin, Jacques Theys, Amaury (Momo) Lendasse have similarly influenced me, and I think I can safely say that I really have enjoyed and benefited from their generosity and collaboration. Momo was the greedy taster of my "Poulet au safran" and "Agneau au coing".

During these four years, I have also benefited much from fruitful exchanges with CESAME visitors, especially Paul's visitors. I would like to thank them all, and particularly Brian Anderson, Thanos Antoulas, Chris Baker, and Danny Sorensen. I would like to thank especially Kyle Gallivan with whom I coauthored my first published paper.

In my first couple of years in CESAME, I benefited from the experience of Benoit Codrons, Benoit David, Frédéric Grognard, Isabelle Motte and Xavier Bombois and many others. I would like to thank everyone who

shared many good times, and introduced me to a world of control, computers and applied mathematics. It has also been my pleasure to work with all other CESAME people, and my roommate Ilyasse. I would like to also thank Axeldyo and his parents for their support and kindness.

Our secretary group (Dominique, Isabelle, Michou and Lydia) is surely the kindest, coolest and most witty persons one could possibly hope to work with. They have also helped me out many a time during those pre-presentation panics. I must also thank them for their unique blend of caring and total organization. I can not imagine CESAME without this group. Several others gave specific technical support, and to all of them I am grateful. I have also a special thought for my friend Victor who left us.

Finally, I would like to thank my parents and my family for their enthusiastic support, and for their continuous love and confidence in my ability to bring this to a good end. I owe them so much, and especially to my mother.

Younès Chahlaoui
Louvain-La-Neuve,
December 9, 2003

Preface

Since computers have been used in engineering, the ability to solve problems numerically increased every day, but despite this evolution, there is still a need for efficient algorithms, especially when dealing with complex systems.
The rapid development of computers in the last 50 years has had an impact on scientific computing whose size is hard to grasp[1]. The interplay between numerical simulations and theoretical models now plays a crucial role in most areas of physics, chemistry, engineering, and other sciences. However, this impact would have been drastically reduced were it not for the parallel development of efficient numerical algorithms. For instance, the development of techniques for sparse matrices [63], the Fast Fourier Transform (see for example *http://www.netlib.org*), and the Fast Multipole Method [25], has allowed scientists to handle hitherto undreamed-of problems.

When handling a real world problem, we have to do three tasks to represent the physical process in mathematical terms. In general, this is done using ordinary differential equations (ODE) or partial differential equations (PDE). The resulting mathematical model is often a continuous problem, in which case one often discretizes it. Actually this is also an order reduction operation as we reduce the continuous problem (infinite dimension) to a matrix problem involving a finite number of variables. Common discretization processes are the finite difference and the finite element methods, which generate very often large sparse matrices (see e.g. [132, 5]). The resulting system is of high complexity since one uses a very fine discretization scheme to have a mathematical system which reflects rigorously the physics of the real world process. But even a small or medium scale system can have a high complexity, especially if we want to deal with it in real time or iterative processing. Notice that by medium scale system we often mean a system of order approximatively one hundred.
In general there are three potential problems with complex systems : storage, computational speed and accuracy. Storage and computational speed are limited so the time needed to do most computations may be prohibitively large. This will affect indirectly the accuracy which depends mainly of problem conditioning. For such cases it is essential to design models of reduced complexity.

In the last twenty years, a variety of model reduction techniques have been developed to produce systematically simpler models of complex systems. A common way to construct a reduced order model is to project the system on a subspace of lower dimension than the dimension of the original system. But in this approach it is crucial to make a good choice of projector in order to have the maximum information of the original system in the projected one.
Projection is often used in everyday life e.g. to make 2-dimensional descriptions of our 3-dimensional world. Consider for example the shadow of a person; it is a projection of the person onto the 2-dimensional surface on which the shadow falls that gives a description of the person. Would we be able to recognize a person

[1] Some idea of the rate of progress of this technology can be gained from such quotes as "Such a machine in the hands of a competent operator can produce 400 full-length products or 1000 sums during an 8-hour working day"[27], referring to an electromechanical desk calculator typical for scientific use in the 1950s. The diagonalization of an 8-by-8 matrix was a weekend-long task[83].

that we know by just looking at his shadow? Probably not, if the camera looks arbitrarily onto the person, but possibly if the camera sees the profile of the face.

Fig. 0.1. Can you recognize the person better from the left or the right picture?

Figure 0.1 shows two pictures of a person[2]. This person is enlightened in the same way. If one know the person, we can recognize him easily from the right picture which represents him better.
The next figure shows the projection of the UCL icon on five different low-rank spaces.

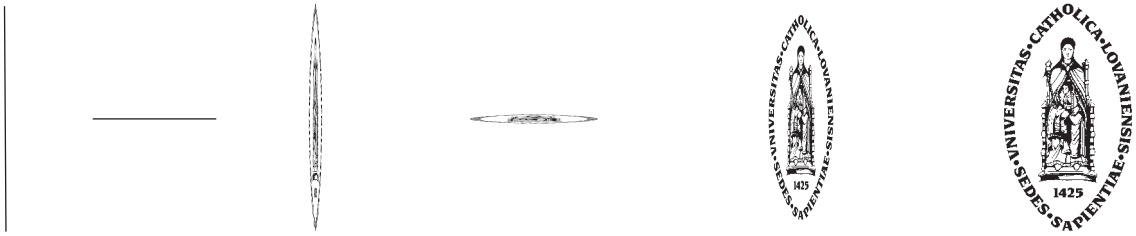


Fig. 0.2. Five projections of the UCL icon.

It can be seen that the first four projections give no idea on the original object which is the UCL icon. The fifth projection gives better information and one can easily recognize it. So the choice of projector is crucial to provide a good approximation.

[2] Thanks to the friendly collaboration of Prof. Paul Van Dooren for this example.

In this dissertation, we are interested in large scale linear dynamical systems, modelled via systems of explicit equations of the type

$$\begin{cases} E_k x_{k+1} = A_k x_k + B_k u_k \\ y_k \quad\quad = C_k x_k. \end{cases}$$

The reduced order model is essentially provided via a projector π of the state-space vector x_k to a vector $\hat{x}_k \doteq \pi x_k$ of much lower dimension.
In general, the objectives in model reduction depend on the intended use of the model and the projector π has to be computed in this perspective. Here, we are mainely interested in finding a smaller model

$$\begin{cases} \hat{E}_k \hat{x}_{k+1} = \hat{A}_k \hat{x}_k + \hat{B}_k u_k \\ \hat{y}_k \quad\quad = \hat{C}_k \hat{x}_k, \end{cases}$$

whose output $\hat{y}_k$ is close to the original output y_k for a given norm. In other words, we are trying to construct a dynamical system of much lower complexity that nevertheless approximates well the behavior of the original system.

A popular model reduction technique for linear time invariant systems (LTI) is Balanced Truncation (BT) [91, 50] for which an a priori error bound on the truncated model can be obtained [61]. This approach "balances out" the states so that each state is as controllable as it is observable; one then just truncates the "weak" states to produce a good reduced order model. In [133, 107] a generalization to linear time-varying systems (LTV) is presented, but without any algorithmic details. Recently, an a priori error bound for time-varying systems that is similar to the time-invariant case, was obtained in [106].
The present thesis focuses on the case were the original system is large and sparse. For such systems it is important to exploit this sparsity for computational efficiency. Iterative methods are very suitable for this and are often easy to parallelize as well. Direct methods like balanced truncation, on the other hand, ignore sparsity in the system and are not very attractive for parallelization : if we denote the order of the system by N, the complexity of these methods is roughly $O(N^3)$ flops and they require about $O(N^2)$ words of memory, which is too expensive computationally to use on large problems.

This work proposes three novel approaches for iterative projection. The main idea is to mix Krylov subspace ideas with the Balanced Truncation procedure to obtain recursively the projector π. This projector is used to provide the reduced order model. The main contribution of this work is the presentation of two algorithms dedicated to recursive model reduction of time-varying systems. They approximate the Gramians and Hankel operators using iterative low-rank approximations which can be updated recursively. The computational cost for constructing these approximations is only of the order of $O(N)$ flops at each iteration, and we obtain an approximation error that is comparable to that obtained via Balanced Truncation. All our results are presented for linear discrete-time systems, but they extend to linear continuous-time systems as well.
The two first algorithms, namely Modified Low-Rank Smith (MLRS) and Recursive Low-Rank Gramian (RLRG), focus on the recursive approximation of the Gramians. Actually, these Gramians are the energy functions of the system. They are solutions of the so-called Stein equations for Linear Time-Invariant (LTI) systems, or are solutions of the Stein recurrences for Linear Time-Varying (LTV) systems.
Using the fact that these Gramians are positive semi-definite, the idea is to approximate recursively the "square root" (or Cholesky factor) of each Gramian by a low-rank approximation. These low-rank versions of the Gramians are then updated recursively using the Singular Value Decomposition (SVD).
This is also the case for the third algorithm, namely Recursive Low-Rank Hankel (RLRH). But, contrary to the two first algorithms which approximate independently the Gramians, the RLRH uses the Hankel map to balance directly the low-rank approximation at each iteration.
The MLRS and RLRG methods provide two low-rank approximations of the original Gramians. These approximated Gramians are used instead of the exact Gramians to provide the projector π via the Balanced Truncation procedure. The RLRH method immediately yields a pair of balanced low-rank approximated Gramians which can be used directly to provide the projector π. This approach is new and recommended for poorly balanced systems.

This combination of the Krylov subspace and the Balanced Truncation approach yields very nice properties. The Krylov subspace ideas lead to iterative computations which reduce significatively the cost and make use of any sparsity in the data. The complexity is linear in the large dimension (i.e. $O(N)$) instead of a cubic complexity (i.e. $O(N^3)$) for Balanced Truncation. The use of the Singular Value Decomposition is borrowed from the Balanced Truncation approach and it yields bounds on the quality of the approximations.

The last major topic of this work is the presentation of model reduction algorithms for second order systems. These systems come mainly from mechanical models and have a special structure. The motivation is to adapt the model reduction methods to this kind of systems, in order to provide a reduced order model which has the same kind of structure. The variables in the reduced model then keep a physical meaning.

The numerical aspects and the efficiency of all our algorithms are assessed by means of a number of real world examples.

Synopsis

This thesis falls naturally into four parts, which are relatively independent.

Part I : Chapter 2 and 3
Part II : Chapter 4 and 5
Part III : Chapter 6 and 7
Part IV : Chapter 8

The first part is constituted by Chapters 2 and 3. Chapter 2 is devoted to the introduction of the preliminary concepts in linear algebra and system theory that will be needed for a better understanding of this thesis. In linear algebra, we mainly review the QR, eigenvalue and singular value decompositions, Krylov spaces, vector and matrix norms. In system theory, we restrict ourselves to the discrete-time linear systems, and we review basics like stability, controllability, observability, and the system Gramians. Most of the material presented here can be found in the literature.
The goal and subject matter of Chapter 3 is to review the state of art for model reduction of linear systems. We review mainly the most used projection based methods in this field, especially Balanced Truncation and Krylov subspaces methods. We also present a new derivation of an a posteriori bound of the $\mathcal{H}_2$ norm of the error system corresponding to the Balanced Truncation method.
We introduce a new method based on a combination of the Krylov subspace and Balanced Truncation procedures. We called it *Approximated Balanced Truncation*. Most techniques for model reduction of linear systems are based on the dominant subspaces of Gramians. In practical applications there is often a rapid decay in the Gramians eigenvalues, and so these Gramians can be well approximated using low-rank approximations. These low-rank approximations are used instead of the original Gramians in the Balanced Truncation procedure to provide the reduced order model. This method has very nice properties depending on how one computes the low-rank approximations of the Gramians.
The three other parts form the main new results of the dissertation, and they are devoted to the presentation and the study of our three recursive low-rank approximation methods, and the adaptation to second order systems.

The second part can be viewed as a description and numerical study of the so-called Modified Low-Rank Smith method (MLRS) (see also [98, 7]).
Chapter 4 presents low-rank approximation methods of the Gramians which can be provided using a Smith like method. In fact, these methods approximate iteratively the "square root" of the Gramians. The best method of this class is the so-called Modified Low-Rank Smith method. It is based on the recursive calculation of dominant singular subspaces. This algorithm is very general and can be applied to many other fields like text retrieval and image compression. We provide in Chapter 5 a detailed description and a numerical study of this recursive procedure. We show also how to use the resulting approximations to produce an Approximated Balanced Truncation reduced model. We illustrate the efficiency of this algorithm using applications to image reconstruction and model reduction. Up to this chapter we only consider linear time-invariant systems.

The goal and subject matter of Part III is the presentation of two novel algorithms mainly dedicated to the time varying case.

Chapter 6 presents a detailed description and numerical study of the Recursive Low-Rank Gramian (RLRG) approximation method. It is also based on an independent low-rank approximation of the Gramians. These approximations are updated recursively using the Singular Value Decomposition.
Chapter 7 presents another approach based on the Hankel map and handles well model reduction of linear systems especially poorly balanced systems. It is called the Recursive Low-Rank Hankel (RLRH) approximation method. We assess the performance of both the RLRG and RLRH methods using some real world benchmark examples.

The fourth and final part is self-contained and discusses model reduction methods for second order systems. This part is connects with other theoretical work in this thesis. It uses ideas of : Balanced Truncation, Krylov subspace, RLRG and RLRH. The goal of Chapter 8 is the adaptation of these methods to second order systems. These systems have a special structure which has some physical meaning. The objective is to preserve this structure in the reduced model, in order to keep the same physical meaning in the reduced order model. This is obtained using methods which we called : Second Order Balanced Truncation (SOBT), Second order Recursive Low-Rank Gramian (SRLRG) and Second order Recursive Low-Rank Hankel (SRLRH). This work has been done in collaboration with Antoine Vandendorpe and Damien Lemonnier.

The second and third parts probably contain the most significant new results; this is reflected in the choice of the thesis title. They are also the main contribution of this thesis.

Citations to previously published work

Chapter 4, and specially Sections. 4.1–4.6, appears in its entirety as

> "Recursive calculation of dominant singular subspaces", Y. Chahlaoui, K. Gallivan and P. Van Dooren, SIAM Journal on Matrix Analysis and Applications, Volume 25, Number 2, pp.445-463, 2003.

and in a short form in

> "Incremental Methods for Computing Dominant Singular Spaces", Y. Chahlaoui, K. Gallivan and P. Van Dooren, book chapter in : 2000 Computational Information Retrieval Workshop, SIAM, Philadelphia, pp.53–62, Ed. M. Berry (2001).

Large portions of Chapters 5, 6, and 7 have appeared in the following papers :

> "Estimating Gramians of large-scale time-varying systems", Y. Chahlaoui and P. Van Dooren, the 15th IFAC World Congress on Automatic Control, Barcelona, July 21-26 , (2002);

> "Recursive Gramian and Hankel map approximation of large dynamical systems", Y. Chahlaoui and P. Van Dooren, The Eighth SIAM Conference on Applied Linear Algebra, LA03, The College of William and Mary, Williamsburg, VA, July 15-19, 2003.
> (Accessible via *http://www.siam.org/meetings/la03/proceedings/*);

> "Recursive low rank Hankel approximation and model reduction", Y. Chahlaoui and P. Van Dooren, Proceedings of the European Control Conference ECC2003, 1-4 September, University of Cambridge, UK, (2003);

> "Second Order Balanced Truncation", Y. Chahlaoui, D. Lemonnier, A. Vandendorpe and P. Van Dooren, submitted to Linear Algebra Appl., Special Issue on Model Reduction, (2003);

> "Model reduction of second order systems", Y. Chahlaoui, D. Lemonnier, K. Meerbergen, A. Vandendorpe and P. Van Dooren, MTNS2002, Fifteenth International Symposium on Mathematical Theory of Networks and Systems, University of Notre Dame, August 12-16, (2002).

We illustrate the quality of our algorithms using numerical examples which were collected in a report of benchmark examples :

> "A collection of benchmark examples for model reduction of linear time invariant dynamical systems ", Y. Chahlaoui and P. Van Dooren, SLICOT Working note 2002-2, NICONET, February (2002).

The NICONET reports can be downloaded as compressed postscript files from the World Wide Web URL : *http://www.win.tue.nl/niconet* and choose : reports or from the WGS ftp site : *ftp://wgs.esat.kuleuven.ac.be* (directory pub/WGS/REPORTS/).

Contents

List of Figures

List of Tables

List of Algorithms

Notation glossary

We describe the abbreviations and symbols used in this dissertation.

Abbreviations

SISO	Single-Input Single-Output
MIMO	Multiple-Input Multiple-Output
SVD	Singular Value Decomposition
C/D	Continuous/Discrete
D/C	Discrete/Continuous
BT	Balanced Truncation
MLRS	Modified Low-Rank Smith
RLRG	Recursive Low-Rank Gramian
RLRH	Recursive Low-Rank Hankel
CGS	Classical Gram-Schmidt
MGS	Modified Gram-Schmidt
flops	floating point operations
CPU	Central Processing Unit
HSV	Hankel Singular Value
LTI	Linear Time Invariant
LTV	Linear Time-Varying

Linear algebra symbols

Generally, in linear algebra[3] we use

capital mathcal letters	$\mathcal{X}$, $\mathcal{V}$	for spaces and subspaces,
capital letters	A, B, Δ, Λ	for matrices,
subscripted lower case letters	a_{ij}, b_{ij}, δ_{ij}, λ_{ij}	for matrix elements,
lower case letters	x, y, v, w	for vectors,
lower case Greek letters	α, β, γ, θ	for scalars.

In addition to that, we will use the following specific notations :

$\mathbb{N}$	group of integers
$\mathbb{C}$	field of complex numbers
$\mathbb{R}$	field of real numbers
$\mathbb{K}$	either $\mathbb{R}$ or $\mathbb{C}$
I_k	denotes the identity matrix[4]
A^T	transpose of A
A^*	complex conjugate transpose of A
A^{-T}	inverse of A^T
A^{-*}	inverse of A^*
$\mathrm{Ker}(A)$	kernel of A
$\mathrm{Im}(A)$	image of A
$rank(A)$	rank of A
$Trace(A)$	trace of A
$\det(A)$	determinant of A
$\kappa(A)$	condition number of A
$\lambda_i(A)$	i-th eigenvalue of A
$\lambda_{\max}(A)$	maximum eigenvalue of A
$\sigma_i(A)$	i-th singular value of A
$\sigma_1(A)$	maximum singular value of A
$\rho(A)$	spectral radius of A, $\rho(A) = \lvert\lambda_{\max}(A)\rvert$
$\lVert\cdot\rVert_2$	2-norm
$\lVert\cdot\rVert_\infty$	infinity norm
$\lVert\cdot\rVert_F$	Frobenius norm
ϵ_m	machine epsilon
l^2_m	space $\{x \in K^m \quad \text{s.t.} \quad \lVert x\rVert_2 < \infty\}$
$K_k(A,V)$	Krylov sequence $(V, AV, \ldots, A^{k-1}V)$
$\mathcal{K}_k(A,V)$	Krylov space $\mathrm{Im}(K_k(A,V))$
$A \otimes B$	Kronecker product of A and B

[3] following the widely used convention introduced by Householder, 1964.

System theory symbols

k_i	initial instant
k_f	final instant
$\mathcal{S}$	original model
T_f	transfer function of $\mathcal{S}$
N	order of $\mathcal{S}$
m	number of inputs
p	number of outputs
x	state vector of $\mathcal{S}$
u	input vector
y	output vector of $\mathcal{S}$
E	descriptor matrix of $\mathcal{S}$
A	state matrix of $\mathcal{S}$
B	input matrix of $\mathcal{S}$
C	output matrix of $\mathcal{S}$
D	feedthrough or direct transmission matrix of $\mathcal{S}$
$\Phi(\cdot,\cdot)$	transition matrix
$H(\cdot,\cdot)$	impulse response
$\mathcal{H}$	Hankel map
$\mathcal{H}(k_f,k,k_i)$	Hankel map at instant k on the finite window $[k_i,k_f]$
T	state transformation
$\|\cdot\|_{\mathcal{H}_2}$	$\mathcal{H}_2$ norm
$\|\cdot\|_{\mathcal{H}_\infty}$	$\mathcal{H}_\infty$ norm

$\mathcal{S}_n$	reduced order model of order n
$\mathcal{S}_e$	error system $\mathcal{S}_e = \mathcal{S} - \mathcal{S}_n$
$\mathcal{S}_n^{BT}$	order n balanced truncated model of $\mathcal{S}$
T_{f_n}	transfer function of $\mathcal{S}_n$
$\hat{x}$	state vector of $\mathcal{S}_n$
$\hat{y}$	output vector of $\mathcal{S}_n$
E_n	descriptor matrix of $\mathcal{S}_n$
A_n	state matrix of $\mathcal{S}_n$
B_n	input matrix of $\mathcal{S}_n$
C_n	output matrix of $\mathcal{S}_n$
D_n	feedthrough or direct transmission matrix of $\mathcal{S}_n$

$\mathcal{G}_c$	controllability Gramian of $\mathcal{S}$
$\mathcal{P}^S$	Smith approximation of $\mathcal{G}_c$
G_c	solution of $AG_cA^T - G_c + BB^T = 0$
$\mathcal{C}_i$	controllability matrix $\left[B\ AB\ A^2B \dots A^{i-1}B\right]$
$\mathcal{C}(k, k_i)$	controllability matrix on the finite window $[k_i, k]$
$\mathcal{C}$	controllability matrix $\left[B\ AB\ A^2B \dots A^{i-1}B \dots\right]$
S	Cholesky factor of $\mathcal{G}_c$
S^{MLRS}	MLRS approximation of S
S_n	RLRG or RLRH approximation of S of rank n
$\mathcal{G}_o$	observability Gramian of $\mathcal{S}$
$\mathcal{Q}^S$	Smith approximation of $\mathcal{G}_o$
G_o	solution of $A^TG_oA - G_o + C^TC = 0$
$\mathcal{O}_i$	observability matrix $\left[C^T\ A^TC^T\ (A^T)^2C^T \dots (A^T)^{i-1}C^T\right]^T$
$\mathcal{O}(k_f, k)$	observability matrix on the finite window $[k, k_f]$
$\mathcal{O}$	observability matrix $\left[C^T\ A^TC^T\ (A^T)^2C^T \dots (A^T)^{i-1}C^T \dots\right]^T$
R	Cholesky factor of $\mathcal{G}_o$
R^{MLRS}	MLRS approximation of R
R_n	RLRG or RLRH approximation of R of rank n
π_r	right projection matrix
π_l	left projection matrix
π	projector $\pi = \pi_r\pi_l^*$
μ or η	noise level
$x_i^{(k)}$	denotes the i^{th} vector during the k^{th} iteration
$x_\bullet$	set of all possible vectors or values x_i, i.e. $x_\bullet \doteq \{x_i\ :\ \forall i\}$
$X_\bullet$	set of all sequence of matrices X_i, i.e. $X_\bullet \doteq \{X_i\ :\ \forall i\}$

Chapter 1

Preliminaries

Before going to the main subject, it is useful to give a general introduction of some fundamental material on linear algebra and linear systems theory that will be necessary for a better understanding of this thesis. Most of the material presented here can be found in the literature. Some new notations are introduced. The discussion is restricted to systems that are finite-dimensional and linear (time-varying or time-invariant).
First, in Section 1.1, we briefly review some important concepts from linear algebra that will be useful in the sequel, like the QR, eigenvalue, singular value decompositions, Krylov spaces, vector and matrix norms. Other definitions and properties will be introduced when needed.
Section 1.2 is devoted to the basics of linear systems theory, including stability, controllability, observability, and the system Gramians as the solutions of two Stein equations. We restrict ourselves to the discrete-time case in most of the text and show briefly how to extend to continuous-time case, using possible conversions.

1.1 Linear algebra

This section summarizes some important matrix properties. For a thorough background of linear algebra the reader is referred e.g. to [63, 69, 47, 45].

Let $\mathbb{K}$ be either $\mathbb{R}$ or $\mathbb{C}$ and let $\mathbb{K}^m$ be the vector space of m-tuples over $\mathbb{K}$. Submatrices are specified with the colon notation, as used in MATLAB : $A(p\!:\!q, r\!:\!s)$ denotes the submatrix of A formed by the intersection of rows p to q and columns r to s. As a special case, a colon only as the row or column specifier means all entries in that row or column. Thus $A(:, j)$ is the jth column of A and $A(i, :)$ the ith row. The values taken by a counter are also described using the colon notation : "$i = 1 : n$" means the same as "$i = 1, 2, \ldots, n$". Also, we describe a set of vectors $x_1, x_2, \ldots, x_n$ by $\{x_i\}_1^n$.

The vectors $\{x_i\}_1^n \in \mathbb{K}^m$ are said to be *linearly dependent* over $\mathbb{K}$ if there exists $\{\alpha_i\}_1^n \in \mathbb{K}$ not all zero so that $\sum_{i=1}^{n} \alpha_i x_i = 0$. Otherwise they are said to be *linearly independent*. The set of all linear combinations of $\{x_i\}_1^n$ is the subspace

$$span\{x_1, \ldots, x_n\} \doteq \{x = \sum_{i=1}^{n} \alpha_i x_i \mid \alpha_i \in \mathbb{K}\}.$$

Let $\mathcal{X}$ be a subspace of $\mathbb{K}^m$, $\{x_i\}_1^n$ is called a *basis* for $\mathcal{X}$ if $\{x_i\}_1^n$ are linearly independent and $\mathcal{X} = span\{x_1, \ldots, x_n\}$. Furthermore $n = dim(\mathcal{X})$ is the *dimension* of $\mathcal{X}$.
Moreover, if we consider the *inner product* (also known as *dot product* or *scalar product*) of x and y denoted by x^*y, the vectors $\{x_i\}_1^n$ are said to be :

- *orthogonal* if $x_i^* x_j = 0,\ \forall i \neq j$;

- *orthonormal* if $x_i^* x_j = \delta_{ij}$, where δ_{ij} is the *Kronecker delta symbol* given by

$$\delta_{ii} = 1, \text{ and } \delta_{ij} = 0, \ \forall j \neq i;$$

- an *orthonormal basis* for $\mathcal{X}$ if they form a basis of $\mathcal{X}$ and are orthonormal.

Consider a matrix[1] $A \doteq [a_{ij}] \in \mathbb{K}^{m \times n}$, we define :

- the *kernel*, $\quad \mathrm{Ker}(A) \doteq \{x \in \mathbb{K}^n \mid Ax = 0\}$;
- the *image*, $\quad \mathrm{Im}(A) \doteq \{y \in \mathbb{K}^m \mid y = Ax, \ x \in K^n\}$,
  note that

$$\mathrm{Im}(A) = span\{A(:,1), \ldots, A(:,n)\};$$

- the *rank*, $\quad rank(A) \doteq dim(\mathrm{Im}(A))$;
- the *transpose*, $\quad A^T \doteq [a_{ji}] \in \mathbb{K}^{n \times m}$;
- the *complex conjugate transpose*, $\quad A^* = [conj(a_{ji})] \in \mathbb{K}^{n \times m}$;
- the *trace* (if $m = n$), $\quad Trace(A) \doteq \sum_{i=1}^{n} a_{ii}$.

The matrix $A \in \mathbb{K}^{n \times n}$ is said to be :

- *symmetric* if $\mathbb{K} = \mathbb{R}$ and $A^T = A$ (notice that then $A^* = A^T$);
- *Hermitian* if $A^* = A$;
- *nonsingular* or *invertible* if the inverse A^{-1} exists,

$$\exists B \in \mathbb{K}^{n \times n} \text{ s.t. } AB = BA = I_n;$$

- *orthogonal* if $\mathbb{K} = \mathbb{R}$ and $A^T = A^{-1}$ (i.e. $AA^T = A^T A = I_n$);
- *unitary* if $A^* = A^{-1}$ (i.e. $AA^* = A^* A = I_n$);
- *positive semi-definite* (respect. *definite*) if it is Hermitian and

$$\forall x \in \mathbb{K}^n \neq 0 \text{ we have } x^* Ax \geq 0 \text{ (respect. } x^* Ax > 0).$$

Two $(n \times n)$-matrices A and $\bar{A}$ are said to be *similar* if there exists a nonsingular matrix T that satisfies $\bar{A} = T^{-1} A T$.

1.1.1 Norms

In this section we define vector and matrix norms. They are very useful to properly define the concepts of energy, action and amplitude of a "signal".

Vector norms

All of the *vector norms* we will use are instances of $p-$norms, which for a real number $p > 0$ and a vector $x \in \mathbb{K}^m$ are defined by

$$\|x\|_p = (\sum_{i=1}^{m} |x_i|^p)^{\frac{1}{p}}.$$

Important special cases are :

[1] In this section we follow the convention to take a rectangular matrix to have dimensions $m \times n$ and $n \times n$ for a square matrix.

$$\text{2-norm} \qquad \|x\|_2 = (x^*x)^{\frac{1}{2}} = (\sum_{i=1}^{m} |x_i|^2)^{\frac{1}{2}},$$

and

$$\infty\text{-norm} \qquad \|x\|_\infty = \max_{1\leq i\leq n} |x_i|,$$

which can be viewed as a limiting case for $p \to \infty$.

Remark 1.1. In signal processing, $\|x\|_\infty$ corresponds to the maximum amplitude of signal x and $\|x\|_2$ to its energy.

All of these norms give similar results qualitatively, but in certain circumstances a particular norm may be easier to work with for analytical or computational reasons. For any vector $x \in \mathbb{K}^m$, we have

$$\|x\|_\infty \leq \|x\|_2 \leq \sqrt{m}\|x\|_\infty,$$

which expresses that these two norms are *equivalent*.

We can now define the following normed space :

$$l_m^2 = \{x \in \mathbb{K}^m \text{ s.t. } \|x\|_2 < \infty\},$$

which is a Hilbert space with inner product $\langle x, y\rangle_{l_m^2} \doteq x^*y$.

Matrix norms

In general, *matrix norms* are defined in terms of an underlying vector norm. Specifically, given a vector norm, we define the corresponding matrix norm of a matrix $A \in \mathbb{K}^{m\times n}$ by

$$\|A\| = \max_{x\neq 0} \frac{\|Ax\|}{\|x\|},$$

such a matrix norm is said to be *induced* by or *subordinate* to the vector norm.
Intuitively, the norm of a matrix measures the maximum stretching the matrix does to any vector, as measured in the given vector norm.
We have

$$\|A\|_2 = \sqrt{\lambda_{max}(A^*A)},$$

where $\lambda_{max}(A^*A)$ is the maximum of the eigenvalues of A^*A (to be defined in (§.1.1.2)),

$$\|A\|_\infty = \max_i \sum_{j=1}^{n} |a_{ij}|.$$

Another useful matrix norm is the so-called *Frobenius norm*. It is defined as

$$\|A\|_F = \sqrt{Trace(A^*A)} = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} |a_{ij}|^2}.$$

However, the Frobenius norm is not an induced norm.

The following properties of matrix norms are easy to show :

Theorem 1.2. *Let A and B be any matrices with appropriate dimensions. Then*

- *$\rho(A) \leq \|A\|$ for any induced norm and for the Frobenius norm (where $\rho(A)$ denote the spectral radius of A (to be defined in (§.1.1.2)));*
- *$\|AB\| \leq \|A\|\|B\|$ for any induced norm;*

- $\|UAV\|_2 = \|A\|_2$, *and* $\|UAV\|_F = \|A\|_F$, *for any appropriately dimensioned unitary matrices* U *and* V*;*
- $\|AB\|_F \leq \|A\|\|B\|_F$ *and* $\|AB\|_F \leq \|B\|\|A\|_F$, *for any induced norm.*

■

For more details on these norms and their properties see for example [63, 115].

1.1.2 Matrix decompositions

Orthogonal transformations and matrix decompositions are very important in numerical linear algebra. They often provide a numerically robust transformation of a given problem to a simpler one whose solution is the "same" as that of the original problem.
The most frequently used transformations are the orthogonal transformation to triangular form which is accomplished by the *QR decomposition*, and the orthogonal transformation to diagonal form which is accomplished by the *eigenvalue* decomposition or the *Singular Value Decomposition* (SVD). We give here a short description of these three important matrix decompositions. For more details we refer to the literature.

QR decomposition

A QR decomposition of $A \in \mathbb{K}^{m\times n}$ with $m \geq n$ is a factorization

$$A = QR = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_1 \\ 0 \end{bmatrix} = Q_1 R_1,$$

where $Q \in \mathbb{K}^{m\times m}$ is orthonormal and $R_1 \in \mathbb{K}^{n\times n}$ is upper-triangular. Depending on the context, either the full factorization $A = QR$ or the "economy size" version $A = Q_1 R_1$ can be called a QR factorization. It can be computed in several ways, including by the use of

- *Gram-Schmidt* orthogonalization : formulated by Gram in 1883 and in its modern algorithmic form by Schmidt in 1907;
- *Givens* transformations : published also in 1958 by Givens, but analyzed in detail by Wilkinson in 1962;
- *Householder* transformations : published by Householder in 1958, his motivation was to compute the QR factorization with less arithmetic operations (in particular, less square roots).

First, the QR decomposition is a reformulation of the Gram-Schmidt orthogonalization process. If we apply Gram-Schmidt to the columns $\{a_j\}_1^n$ of A from left to right, we get a sequence of orthonormal vectors $\{q_j\}_1^n$ spanning the same space : these orthogonal vectors are the columns of Q. Gram-Schmidt also computes coefficients $r_{ji} = q_j^* a_i$ expressing each column a_i as a linear combination of $\{q_j\}_1^i$: $a_i = \sum_{j=1}^{i} r_{ji} q_j$. The r_{ji} are just the entries of R.
There exist two versions of the Gram-Schmidt method, which are shown in the following algorithms.

Algorithm 1 The Classical Gram-Schmidt algorithm (CGS).

Ensure: Given $A \in R^{m\times n}$ of rank n this algorithm computes the QR decomposition $A = QR$, where Q is $m \times n$ and R is $n \times n$, by the Gram-Schmidt method.

```
for i = 1 : n do
  q_i = a_i;
  for j = 1 : (i − 1) do
    r_ji = q_j^T a_i;
    q_i = q_i − r_ji q_j;
  end for
  r_ii = ‖q_i‖_2;
  if r_ii = 0 then
    break {a_i is linearly dependent on a_{1:i−1}}
  end if
  q_i = q_i / r_ii;
end for
```

Unfortunately, the CGS algorithm is numerically unstable when the columns of A are nearly linearly dependent. The method can be rearranged so that as soon as q_i is computed, all the remaining vectors are orthogonalized against q_i. This gives the modified Gram-Schmidt algorithm.

Algorithm 2 The Modified Gram-Schmidt algorithm (MGS).

Ensure: Given $A \in R^{m\times n}$ of rank n this algorithm computes the QR decomposition $A = QR$, where Q is $m \times n$ and R is $n \times n$, by the MGS method.

```
for i = 1 : n do
  r_ii = ‖a_i‖_2;
  if r_ii = 0 then
    break {Stop if linearly dependent.}
  end if
  q_i = a_i / r_ii;
  for j = (i + 1) : n do
    r_ij = q_i^T a_j;
    a_j = a_j − r_ij q_i;
  end for
end for
```

Unlike the CGS algorithm, the MGS algorithm permits the use of column pivoting to identify a maximal linearly independent set of columns of A. We can see easily that if A has full column rank, r_{ii} will not be zero. Björk [23, 24] has shown that the accuracy and the loss of orthogonality for the MGS algorithm are as follow :

$$\|A - QR\| \approx \epsilon_m \|A\|,$$
$$Q^T Q - I_n = E_{MGS}, \qquad \|E_{MGS}\|_2 \approx \epsilon_m \kappa(A),$$

where ϵ_m denote the epsilon machine (see §. 1.1.5). We mention that the CGS algorithm has the same order of accuracy, but unfortunately the loss of orthogonality is worse (see [69, 63]), i.e.,

$$\|A - QR\| \approx \epsilon_m \|A\|,$$
$$Q^T Q - I_n = E_{CGS}, \qquad \|E_{CGS}\|_2 \approx \epsilon_m \kappa(A)^2.$$

This residual bound for CGS was proved only for a special case, and it does not hold in general [69].

We can also use Givens rotations to introduce zeros into vectors that they multiply. A Givens rotation $G(i,j,\theta) \in \mathbb{K}^{n\times n}$ is equal to the identity matrix except for the 2×2 matrix

$$G([i,j],[i,j]) = \begin{bmatrix} c & s \\ -s & c \end{bmatrix},$$

where $c = \cos\ \theta$ and $s = \sin\ \theta$. The multiplication $y = G(i,j,\theta).x$ rotates x through θ radians clockwise in the (i,j) plane. Algebraically,

$$y_k = \begin{cases} x_k, & k \neq i,j, \\ cx_i + sx_j, & k = i, \\ -sx_i + cx_j, & k = j, \end{cases}$$

and so $y_j = 0$ if

$$s = \frac{x_j}{\sqrt{x_i^2 + x_j^2}}, \quad c = \frac{x_i}{\sqrt{x_i^2 + x_j^2}}.$$

Givens rotations are therefore useful for introducing zeros into a vector one at a time. To compute the QR factorization, Givens rotations are used to eliminate the elements below the diagonal in a systematic fashion. We will see later how to use these decompositions in the cases where we are interested in with more details.

Another way to annihilate desired components of a given vector is the Householder transformation, which is a matrix of the form

$$H = I - 2\frac{vv^T}{v^T v},$$

where v is a nonzero vector. We can easily see that H is both orthogonal and symmetric (i.e., $H = H^T = H^{-1}$). The QR decomposition can be computed using a suitable sequence of Householder matrices [63]. The general process can be described as follows [69]. With $A_1 = A$ we have, at the start of the i^{th} stage

$$A_i = \begin{bmatrix} R_{i-1} & z_i & B_i \\ 0 & x_i & C_i \end{bmatrix}, \quad R_{i-1} \in \mathbb{R}^{(i-1)\times(i-1)}, \quad x_i \in \mathbb{R}^{m-i+1},$$

where R_{i-1} is upper triangular. Choose a Householder matrix $\tilde{H}_i$ such that $\tilde{H}_i x_i = \alpha e_1$ and embed $\tilde{H}_i$ into an $m \times m$ matrix

$$H_i = \begin{bmatrix} I_{i-1} & 0 \\ 0 & \tilde{H}_i \end{bmatrix}.$$

Then let $A_{i+1} = H_i A_i$. Overall, we obtain $R = H_n H_{n-1} \dots H_1 A =: Q^T A$ ($H_n = I$ if $m = n$). The Householder transformations are very suitable for parallel computations and are more robust.
The accuracy and the loss of orthogonality for both the Givens and Householder approaches are of the form :

$$\|A - QR\| \approx \epsilon_m \|A\|,$$
$$Q^T Q - I_n = E, \qquad \|E\|_2 \approx \epsilon_m,$$

and hence are better than the Gram-Schmidt approach.

Eigenvalues and eigenvectors

One of the oldest matrix decompositions is the *eigendecomposition*. The concept of *eigenvalues* predates the formal notion of matrices, which was introduced by Cayley in 1855. The name eigenvalues did not become standard, however, until the mid twentieth century; previously they had been called *characteristic values*, *proper values*, or *latent roots*. The notion of eigenstructure can be traced back as far as Cauchy, but clearly appears in the work of Jacobi (1845), Weierstrass (1868) and Schur (1909). But, practical methods for computing these eigenvalues appear only in the twentieth century, for example by Muntz (1913), Krylov (1931), Wielandt (1944), Lanczos (1950), Arnoldi (1951), and others (see e.g. [135]).

Let $A \in \mathbb{K}^{n\times n}$, then the eigenvalues of A are the n roots of its *characteristic polynomial* $p_A(\lambda) \doteq \det(\lambda I - A)$. This set of roots is called the *spectrum* of A and is denoted by $\Lambda(A)$. That is, $\Lambda(A) \doteq \{\lambda_1, \ldots, \lambda_n\}$ if λ_i are the roots of $p_A(\lambda)$, multiplicities counted. The maximal modulus of the eigenvalues is called the *spectral radius*, denoted by

$$\rho(A) \doteq \max_{1\leq i\leq n} |\lambda_i|,$$

where, as usual, $|\ .\ |$ denotes the magnitude.
If $\lambda \in \Lambda(A)$ then any nonzero vector $v \in K^n$ that satisfies

$$Av = \lambda v,$$

is referred to as a *right eigenvector* of A. By duality, a nonzero vector w is called a *left eigenvector* of A if

$$w^T A = \lambda w^T.$$

Now, if the matrix A has a complete system of *eigenpairs* (λ_i, v_i), we can combine all these formulas in one matrix equation, and we obtain

$$A = V\Lambda V^{-1},$$

where $V = (v_1, \ldots, v_n)$ and $\Lambda = diag(\lambda_1, \ldots, \lambda_n)$.
This decomposition can be related to another interesting decomposition that we describe below.

Singular value decomposition

The *Singular Value Decomposition* (SVD) is a matrix factorization whose computation is an important step in most low-rank approximation problems.
It was discovered independently by Beltrami (1873) and Jordan (1874) and again by Sylvester (1889). Related work was done by Autonne (1915), Tagaki (1925), Williamson (1935), and others. For a detailed history of the SVD, see [114, 70]. The SVD has a wide variety of applications, including e.g. signal processing [124], control [91, 127], and information retrieval [21]. The SVD has many important geometric and analytic motivations see e.g. [122, 126].
Despite these deep roots, the SVD did not become widely known in applied mathematics until in the late 1960s, Golub and others showed that it could be computed efficiently and used as the basis for many stable algorithms. Even after this time, the mathematical world was slow to recognize the fundamental properties of the SVD (stability and analytic properties) in contrast with the eigenvalues which have been appreciated for their algebraic properties from the beginning.

We now give a formal definition of the SVD. Given a matrix $A \in \mathbb{C}^{m\times n}$, a singular value decomposition of A is a factorization

$$A = U\Sigma V^*$$

where $U \in \mathbb{C}^{m\times m}$, and $V \in \mathbb{C}^{n\times n}$ are unitary, and $\Sigma \in \mathbb{R}^{m\times n}$ is diagonal. In addition, the diagonal entries σ_i of Σ are ordered such that

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_p \geq 0, \quad \text{where} \quad p = \min(m, n).$$

The columns u_i of U and v_i of V are the corresponding left and right singular vectors. This version of SVD is overdetermined. There is a reduced "economy size" SVD, called the *Short Singular Value Decomposition* (SSVD)

$$A = U_1 \Sigma_1 V_1^*$$

where

$$U_1 \in \mathbb{C}^{m \times r}, \quad V_1 \in \mathbb{C}^{r \times r}, \quad \Sigma_1 \in \mathbb{R}^{r \times r} \quad \text{and} \quad r = rank(A).$$

The SVD is closely related to another traditional matrix factorization namely the eigenvalue decomposition. Indeed, σ_i^2 is an eigenvalue of AA^* or A^*A, u_i is an eigenvector of AA^*, and v_i is an eigenvector of A^*A,

$$AA^*u_i = \sigma_i^2 u_i, \quad A^*Av_i = \sigma_i^2 v_i.$$

Using these vectors we can rewrite the matrix A as the sum of r rank-one matrices :

$$A = \sum_{i=1}^{r} \sigma_i u_i v_i^*.$$

Using the SVD we can verify easily some norm equalities :

$$\|A\|_2 = \sigma_1(A),$$

$$\|A\|_F = \sum_{i=1}^{r} \sigma_i(A).$$

An important application of the SVD is the connection with low-rank approximation. It gives an answer to a classical question : what is the rank ν approximation with $(0 \leq \nu \leq r)$ which captures as much of the energy of A as possible?. We can make it precise by formulating a problem of best approximation of a matrix A by matrices of lower rank. This result is known as the theorem of Schmidt-Mirsky (or Eckart-Young).

Theorem 1.3. *For any ν with $(0 \leq \nu \leq r)$, define*

$$A_\nu = \sum_{i=1}^{\nu} \sigma_i u_i v_i^*,$$

if $\nu = p = \min(m, n)$, define $\sigma_{\nu+1} = 0$. Then

$$\|A - A_\nu\|_2 = \inf_{\substack{B \in \mathbb{C}^{m \times n} \\ rank(B) \leq \nu}} \|A - B\|_2 = \sigma_{\nu+1},$$

and

$$\|A - A_\nu\|_F = \inf_{\substack{B \in \mathbb{C}^{m \times n} \\ rank(B) \leq \nu}} \|A - B\|_F = \sqrt{\sigma_{\nu+1}^2 + \ldots + \sigma_r^2}.$$

■

This idea has ramifications in diverse areas like image compression and functional analysis (see e.g. [114, 40, 115, 42]).

1.1.3 Krylov spaces

Most iterative methods for large linear algebra problems are based on the recursive generation of the so-called Krylov spaces of ever increasing dimensions, and the projection of the original matrix operator(s) onto these Krylov spaces [63]. The success of these algorithms comes from the fact that as the Krylov space dimension increases, the new projection can be obtained at low cost from the previous projection onto the next lower dimensional Krylov space.
It is only recently that these techniques have been applied to control problems of large dimension and particulary in the domain of model reduction. In this field, the matrices of a model can be projected in this way. In fact this is the idea behind most of the iterative model reduction methods. It has been shown that the convergence rate is faster than that which would be expected from other methods. Another advantage of Krylov based methods is that they take advantage of sparsity of the model matrices [103]. In this section, we define a Krylov space and review some of the basic algorithms for computing bases for the Krylov space and the projection of a given matrix operator onto these spaces.

In 1931 Krylov used the sequence generated by the power method (now called a Krylov sequence) to determine the characteristic polynomial of a matrix. A Krylov sequence is generated as follows by a matrix $A \in \mathbb{C}^{n\times n}$ and a matrix $V \in \mathbb{C}^{n\times p}$

$$K_k(A,V) \equiv (V, AV, A^2V, \ldots, A^{k-1}V),$$

and the corresponding column space is called the k-th Krylov space and is denoted by $\mathcal{K}_k(A,V)$, i.e.,

$$\mathcal{K}_k(A,V) = \mathrm{Im}(K_k(A,V)).$$

As the vectors $V, AV, A^2V, \ldots$ are generated and appended to form increasing Krylov sequences, the following theorem states that the rank strictly increases at every step until it reaches a maximal value. In fact, the maximal rank is achieved in at most n steps.

Theorem 1.4. *[115]*

$$rank\ K_k(A,V) = rank\ K_{k+1}(A,V)$$

iff

$$rank\ K_k(A,V) = \max_i\ rank\ K_i(A,V) \equiv rank\ K_n(A,V) \equiv rank\ K_\infty(A,V)$$

Furthermore, the space $\mathcal{K}_\infty(A,V)$ is invariant[2] *under A.* ■

There is a large amount of literature on Krylov methods for solving linear systems. We only mention the following references for more details [11, 63, 103].

The most used algorithms for recursively generating bases for the Krylov spaces are :

- The *Arnoldi algorithm* [9] which for a given matrix and vector pair, $\{A, v_1\}$, simultaneously generates orthonormal bases for the Krylov subspaces, $\mathcal{K}_k(A, v_1)$, associated with that pair, and Hessenberg matrices H_k which are matrix representations of the orthogonal projections of A onto the Krylov subspaces. Theoretically, each step of the Arnoldi recursion is well-defined. However, at each step in the Arnoldi recursion all of the previously-generated Arnoldi basis vectors must be kept in storage.
- The *Lanczos algorithm* [83] which consists of two recursions. For a given matrix and vector triplet, $\{A, v_1, w_1\}$, the Lanczos recursion simultaneously generates bi-orthogonal bases for the Krylov subspaces $\mathcal{K}_k(A, v_1)$ and $\mathcal{K}_k(A^T, w_1)$, and tridiagonal matrices T_k which are matrix representations of the bi-orthogonal projections of A onto these Krylov subspaces. These recursions are an implementation of a two-sided Gram Schmidt orthogonalization. Therefore, there is no guarantee that they will not break down. However, in contrast with the Arnoldi procedure, at any step in Lanczos recursion only a few of the most recently-generated Lanczos vectors must be kept in storage.
  When A is reel and symmetric and $w_1 = v_1$, then the Lanczos recursion is equivalent to the Arnoldi recursion.

[2] i.e., $\forall v \in \mathcal{K}_\infty(A,V)$ we have $Av \in \mathcal{K}_\infty(A,V)$.

These two algorithms are often related to the minimization of an objective function, which yields two useful methods :

- The *Generalized Minimal RESidual* method (GMRES) [104] which picks the "best" solution x_k ($k \geq 1$) in the Krylov space $\mathcal{K}_k(A, v)$. "Best" means that the residual $\|v - Ax\|_2$ is as small as possible over $\mathcal{K}_k(A, v)$. It uses the Arnoldi algorithm and so the resulting basis for the Krylov subspace is orthogonal [14].
- The *Quasi-Minimal Residual* method (QMR) [52] which solves the system in a least squares sense, similar to GMRES. But as this method uses the Lanczos algorithm the constructed basis for the Krylov subspace is bi-orthogonal, rather than orthogonal as in GMRES [14].

More recently, many modifications and connections were described in the literature allowing these algorithms to handle all kind of situations (see e.g. [26] for a survey). We will later see how to use the idea of Krylov methods in model reduction.

1.1.4 Kronecker product

Given $A \in \mathbb{C}^{m\times n}$ and $B \in \mathbb{C}^{p\times q}$, the Kronecker product of A and B is defined by

$$A \otimes B \doteq [a_{ij}B] = \begin{bmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \dots & a_{mn}B \end{bmatrix} \in \mathbb{C}^{mp\times nq}.$$

With any matrix X we can associate a vector $vec(X)$:

$$vec(X) \doteq \begin{bmatrix} x_{11} \\ \vdots \\ x_{m1} \\ x_{12} \\ \vdots \\ x_{m2} \\ \vdots \\ x_{1n} \\ \vdots \\ x_{mn} \end{bmatrix} \in \mathbb{C}^{mn}.$$

Then we have the following useful identity :

$$vec(AXB) = (B^T \otimes A)vec(X),$$

where X is any matrix in $\mathbb{C}^{n\times p}$. Furthermore, if A and B are square (i.e., $m = n$ and $p = q$) the eigenpairs of $A \otimes B$ are $(\lambda_i\mu_j, x_i \otimes y_j)$, $i = 1 : n,\ j = 1 : p$, where (λ_i, x_i) and (μ_j, y_j) are the eigenpairs of A and B, respectively.
The Kronecker product is very useful for solving Stein and Sylvester equations. We will see this in more detail in (§.1.2.7).

1.1.5 Complexity and computer arithmetic

We measure the cost of algorithms in *flops*. A flops is an elementary floating-point operation : $+$, $-$, $/$, or $*$. In quantifying operation counts and the accuracy of approximations, we will often use the O notation to indicate the order of magnitude, or dominant term, of a function. For an operation count, we are interested in the behavior as the order of the problem, say n, becomes large. We say that

$$f(n) = O(g(n))$$

(read "f is of the order of g") if there is a positive constant C so that

$$|f(n)| \leq C|g(n)|$$

for all n sufficiently large. For example,

$$5n^3 + 23n^2 + 10n = 5n^3 + O(n^2)$$

because as n becomes large, the terms of order lower than n^3 become relatively insignificant.

Today, IEEE standard floating-point arithmetic is used for the majority of scientific computations [69]. It is built into the hardware of almost all modern microprocessors, and includes the definition of unit roundoff (machine epsilon) which in double precision is $\epsilon_m = 2^{-53} \approx 1.1 \times 10^{-16}$.

In Table 1.1, we list the amount of work required by some matrix decompositions given in (§.1.1.2) [63], where we only neglected the lower order terms.

SVD	U, Σ, V	$14m^2n + 8mn^2 + 9n^3$
SSVD	U_1, Σ, V	$14mn^2 + 8n^3$
QR	Householder	$2mn^2 + 2n^3/3$
	Givens	$3mn^2 + 3n^3/3$
	MGS and CGS	$2mn^2$

Table 1.1. Cost of some matrix decompositions.

All our numerical experiments were carried out using one of the most widely used computing environments MATLAB, which is a commercial product of The MathWorks, Inc. (www.mathworks.com). MATLAB, which stands for MATrix LABoratory, is an interactive system that integrates extensive mathematical capabilities, especially in linear algebra, with powerful scientific visualization, a high-level programming language, and a variety of optional "toolboxes" that provide specialized capabilities in particular applications, such as signal processing, image processing, control, system identification, optimization, and statistics. There exist similar packages that are freely available via the World-Wide Web like SCILAB (www-rocq.inria.fr/scilab).

The Control Toolbox of MATLAB is not very attractive for large scale computations as it uses dense complex matrices as the main data structure, which does not allow to exploit the structure in certain matrices. Instead, it is recommended to use SLICOT, a freeware fortran77 subroutine library based on numerical linear algebra routines from BLAS and LAPACK libraries. It is freely downloadable via *www.win.tue.nl/niconet/niconet.html*. SLICOT provides software for the design and analysis of control systems. SLICOT routines are linked to MATLAB through a gateway compiler and one can use that software in a similar manner as the usual MATLAB m-files. The performance of SLICOT has been assessed with respect to numerical quality, computational speed, and memory requirements for a variety of situations and it has been shown [108] that the integrated routines have significantly improved performance over those of the MATLAB Control Toolbox.

1.2 System theory

A dynamical system is a well defined mathematical object for which many descriptions are possible. The most frequently used is the systems approach which is characterized by the notion of input, state and output, determined on some time domain. In the real world, most systems are nonlinear but, in general, their analysis is difficult and moreover beyond the scope of our interest here. We will therefore focus on linear systems. There are many different ways of describing linear systems, but we limit ourselves to :

1. state space description,
2. input-output description in the time domain.

The choice of any of these two approaches depends on the user and what data are at hand. For large systems, one is often more interested in the macroscopic behavior of a system, and less concerned with the internal model details. An input-output description may then be more suited to describe its behaviour.

1.2.1 State space description

The following explicit nonlinear system describes a quite general type of discrete-time system. The time varying input u_k drives a system $\mathcal{S}$ with state x_k giving output signal y_k, and the system dynamics have the form

$$\begin{cases} x_{k+1} = f(x_k, u_k, k), \\ y_k \ \ = g(x_k, u_k, k). \end{cases} \tag{1.1}$$

The first equation is the *state equation* and the second one is the *output equation*. Such systems can be represented by a block diagram as shown in Figure 1.1.

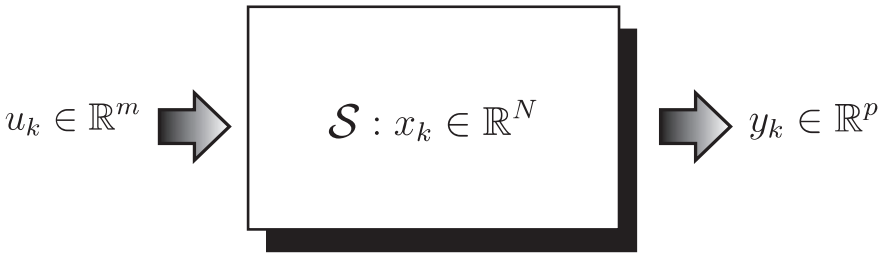


Fig. 1.1. A general block diagram of a dynamical system.

Together with some initial condition (typically $x_0 = 0$), the functions f and g in (1.1) define an input-output map $\mathcal{H}$ taking $u_\bullet$ to $y_\bullet$ [3]. Since this is the relation we are concerned with, rather than the system's internal dynamics, the input-output behaviour is what we will focus on. We first examine these for discrete-time systems and will later briefly address the continuous-time case (§.1.2.9).

For linear discrete-time-varying systems, the state equation and output equation may be simplified to

$$\begin{cases} x_{k+1} = A_k x_k + B_k u_k, \\ y_k \ \ = C_k x_k + D_k u_k, \end{cases} \tag{1.2}$$

where

$x_k \in \mathbb{K}^N$ is the state vector,
$y_k \in \mathbb{K}^p$ is the output vector,
$u_k \in \mathbb{K}^m$ is the input vector,
$A_k \in \mathbb{K}^{N\times N}$ is the state matrix,
$B_k \in \mathbb{K}^{N\times m}$ is the input matrix,
$C_k \in \mathbb{K}^{p\times N}$ is the output matrix,
$D_k \in \mathbb{K}^{p\times m}$ is the direct transmission matrix.

[3] Here we use the notation $m_\bullet$ to denote the whole sequence $\{m_i\}_i$, i.e.

$$m_\bullet \doteq \{m_i \ : \ \forall i\}.$$

The appearance of the variable k in the arguments of matrices A_k, B_k, C_k and D_k implies that these matrices are time-varying. If the variable k does not appear explicitly in the matrices, they are assumed to be time-invariant, or constant. That is, if the system is time-invariant, then the last two equations can be simplified to

$$\begin{cases} x_{k+1} = Ax_k + Bu_k, \\ y_k \quad = Cx_k + Du_k. \end{cases} \tag{1.3}$$

In this case, using the z-transform[4], we can define the transfer function

$$T_f(z) \doteq C(zI - A)^{-1}B + D.$$

Sometimes we have to consider the *generalized* state-space version (called also *descriptor system*), namely

$$\begin{cases} Ex_{k+1} = Ax_k + Bu_k, \\ y_k \quad\quad = Cx_k + Du_k, \end{cases} \tag{1.4}$$

which gives the transfer function

$$T_f(z) \doteq C(zE - A)^{-1}B + D.$$

Here, if the matrix E is nonsingular we can rewrite (1.4) in the form of (1.3). But, in general, to avoid possible numerical problems we try not to invert E even in the case where it is feasible. We will use the descriptor system if it is necessary to give a more general analysis, otherwise we use the simplified version (1.3) which is known as the "standard" state-space description.

Linear discrete-time systems

For time varying systems, the input sequence is generally assumed to be square-summable, i.e $u_k \in l_2^m$. For infinite horizon problems we assume that $\{A_k\}_0^\infty$, $\{B_k\}_0^\infty$, and $\{C_k\}_0^\infty$ are bounded[5] sequences of matrices with appropriate dimensions.
Using the recurrence (1.2) over several time steps, one obtains the state at step k as a function of past inputs over the interval $[k_i, k-1]$:

$$x_k = \Phi(k, k_i)x_{k_i} + \sum_{i=k_i}^{k-1} \Phi(k, i+1)B_i u_i$$

where

$$\Phi(k, k_i) \doteq A_{k-1} \dots A_{k_i}$$

is the discrete transition matrix over the time period $[k_i, k-1]$. The transition matrix has the following properties :

$$\begin{array}{rll} \Phi(k+1, k_i) = A_k\Phi(k, k_i), & & k_i \le k, \\ \Phi(k_2, k_i) \quad = \Phi(k_2, k_1)\Phi(k_1, k_i), & & k_i \le k_1 \le k_2, \\ \Phi(k, k) \quad = I_N, & & \forall k. \end{array}$$

Remark 1.5. It is only when A_k is nonsingular $\forall k \in \mathbb{N}$, that the first property can be solved for $\Phi(k, k_i)$ in terms of $\Phi(k+1, k_i)$. In this case, $\Phi(\cdot, \cdot)$ is defined on all of $\mathbb{N} \times \mathbb{N}$.

[4] The term z correspond to the "advanced" version of $x_\bullet$, i.e., $zx_k = x_{k+1}$.
[5] A sequence of matrices $\{M_k\}$ is said to be bounded if there exists a constant $M \in \mathbb{R}$ so that $\|M_k\| \le M$, $\forall k \in \mathbb{Z}$

One notable special case of time varying systems is the case of a periodic system with period τ, for which the matrices A_k, B_k, C_k and D_k satisfy

$$\begin{array}{l} A_{k+\tau} = A_k, \\ B_{k+\tau} = B_k, \\ C_{k+\tau} = C_k, \\ D_{k+\tau} = D_k, \end{array} \quad \forall k \in \mathbb{N}, \quad \text{and} \quad \det A_k \neq 0. \tag{1.5}$$

In this case the transition matrix has a very nice property : there exists a periodic matrix Ψ_k, called the *monodromy* matrix, such that

$$\Phi(k+\tau, k_i) = \Psi_k \Phi(k, k_i),$$

whence for $k = k_i$

$$\Phi(k+\tau, k) = \Psi_k.$$

For periodic systems there exists a state transformation which yields a time-invariant version of the system,

$$\left\{ \begin{array}{rl} \hat{x}_{k+1} &= \hat{A}\hat{x}_k + \hat{B}\hat{u}_k \\ \hat{y}_k &= \hat{C}\hat{x}_k + \hat{D}\hat{u}_k \end{array} \right., \tag{1.6}$$

where the system matrices can be defined in different manners [113, 123, 121, 28] (For a survey of computational methods for periodic systems see e.g. [131]). A very convenient "lifted" time-invariant system can be defined using the matrices

$$\hat{A} = \begin{bmatrix} 0 & \dots & 0 & A_{\tau-1} \\ A_0 & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \vdots \\ 0 & \dots & A_{\tau-2} & 0 \end{bmatrix}, \quad \hat{B} = \begin{bmatrix} B_0 & & & \\ & B_1 & & \\ & & \ddots & \\ & & & B_{\tau-1} \end{bmatrix},$$

$$\hat{C} = \begin{bmatrix} 0 & \dots & 0 & C_{\tau-1} \\ C_0 & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \vdots \\ 0 & \dots & C_{\tau-2} & 0 \end{bmatrix}, \quad \hat{D} = \begin{bmatrix} D_0 & & & \\ & D_1 & & \\ & & \ddots & \\ & & & D_{\tau-1} \end{bmatrix},$$

and the states, inputs and outputs are "stacked" as follows

$$\hat{x}_k = \begin{bmatrix} x_{k\tau} \\ x_{1+k\tau} \\ \vdots \\ x_{(k+1)\tau-1} \end{bmatrix}, \quad \hat{y}_k = \begin{bmatrix} y_{k\tau} \\ y_{1+k\tau} \\ \vdots \\ y_{(k+1)\tau-1} \end{bmatrix}, \quad \text{and} \quad \hat{u}_k = \begin{bmatrix} u_{k\tau} \\ u_{1+k\tau} \\ \vdots \\ u_{(k+1)\tau-1} \end{bmatrix}.$$

In the situations that we are interested in, the direct transmission matrices[6] $D_\bullet$ do not play any role for model reduction methods that we are interested in (§.1.2.3). So from now on, we will assume $D_\bullet = 0$.

[6] Here also, we use the notation $M_\bullet$ which denote the whole sequence of matrices $\{M_i\}_i$, i.e.

$$M_\bullet \doteq \{M_i \ : \ \forall i\}.$$

1.2.2 Input-output description

An input-output (I/O) description of a linear discrete-time system is obtained using a linear transformation $\mathcal{H}$ which maps an input sequence into an output sequence, $\mathcal{H}$ is therefore called the *I/O map* or the *Hankel map*. It is obtained using the impulse response $H(k,i)$, defined below. Using the realization theory we can make the link between this description and the state-space description [48], and we obtain for the impulse response $H(k,i)$ the matrix-valued sequence given by

$$H(k,i) \doteq \begin{cases} C_k \Phi(k,i+1) B_i & \text{for} \quad k_i \leq i < k, \\ 0 & \text{for} \quad i < k_i. \end{cases} \tag{1.7}$$

This impulse response is restricted to the finite window $[k_i, k_f]$ by restricting the inputs to be non-zero only in the interval $[k_i, k-1]$ (i.e., the "past") denoted $u_{[k_i,k-1]}$, and considering the outputs y_i for $i \geq k$ (i.e., the "future") denoted $y_{[k,\bullet)}$ (see Figure 1.2).

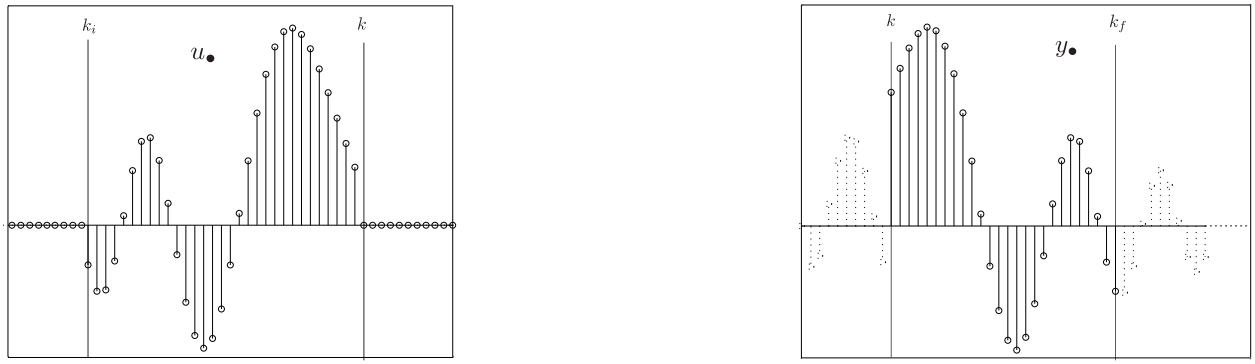


Fig. 1.2. Mapping inputs $u_{[k_i,k-1]}$ to outputs $y_{[k,\bullet)}$.

We then obtain

$$y_k = \mathcal{H}[u]_k \doteq \sum_{i=k_i}^{k} H(k,i) u_i, \quad \forall k \geq k_i. \tag{1.8}$$

Note that in the linear time-invariant case $\mathcal{H}$ is the convolution of the impulse response $H(.)$ with the input $u_\bullet$

$$y_k = \mathcal{H}[u]_k = \sum_{i=0}^{k} H(k-i) u_i = (H * u)_k, \quad \forall k \geq 0, \tag{1.9}$$

where

$$H(k) \doteq H(k,0) = \begin{cases} C A^{k-1} B & \text{for} \quad k > 0 \\ 0 & \text{for} \quad k \leq 0 \end{cases}. \tag{1.10}$$

1.2.3 Similarity transformation

Consider the dynamics (1.3) under a change-of-basis $z_k = T_k x_k$, where T_k is invertible and

$$\|T_k\|, \ \|T_k^{-1}\| \leq M, \quad \text{independently of } k.$$

The new dynamics may then be written as

$$\begin{cases} z_{k+1} = T_{k+1}A_kT_k^{-1}z_k + T_{k+1}B_ku_k \\ y_k \;\; = C_kT_k^{-1}z_k + D_ku_k \end{cases} \tag{1.11}$$

Thus changing the basis of the state space defines a mapping on the state space models given by

$$\left[\begin{array}{c|c} A_k & B_k \\ \hline C_k & D_k \end{array}\right] \quad \rightarrow \quad \left[\begin{array}{c|c} T_{k+1}A_kT_k^{-1} & T_{k+1}B_k \\ \hline C_kT_k^{-1} & D_k \end{array}\right] = \left[\begin{array}{c|c} \hat{A}_k & \hat{B}_k \\ \hline \hat{C}_k & D_k \end{array}\right].$$

(Note that D_k is unchanged and so we can assume without loss of generality that $D_\bullet = 0$). Such maps are called *similarity* or *Lyapunov* transformations. Because these transformations are merely a rewriting of the system dynamics, the input-output map remains the same.

$$\hat{C}_k\hat{\Phi}(k, i+1)\hat{B}_i = C_k\Phi(k, i+1)B_i,$$

where $\hat{\Phi}(k, i+1)$ is the corresponding transition matrix to the transformed system

$$\hat{\Phi}(k, i+1) = \hat{A}_{k-1}\hat{A}_{k-2}\dots\hat{A}_{i+1},$$

for all k and i so that $k > i$. Note that for all j, $i+1 \leq j \leq k$

$$\begin{aligned} \hat{\Phi}(k, j) &= \hat{A}_{k-1}\hat{A}_{k-2}\dots\hat{A}_j \\ &= T_kA_{k-1}T_{k-1}^{-1}.T_{k-1}A_{k-2}T_{k-2}^{-1}\dots T_{j+1}A_jT_j^{-1} \\ &= T_k \underbrace{A_{k-1}A_{k-2}\dots A_j}_{\Phi(k,j)} T_j^{-1} \\ &= T_k\Phi(k, j)T_j^{-1}. \end{aligned}$$

As a given state space model is not a unique description of an input-output map $\mathcal{H}$, the Lyapunov or similarity transformations can be used to find for example :

- the lowest order model with the same $\mathcal{H}$ as the given model, this lowest order model, called the minimal realization, is obtained using the notions of controllability and observability Gramians (§.1.2.5), this minimal realization is non unique;
- a *balanced* realization, for which the Gramians satisfy some additional equation that lead to some nice properties (see Balanced truncation) (§.2.2).

Let us now see some important notions about dynamical systems like : *stability*, *controllability*, *observability* and system characteristics.

1.2.4 Stability

We consider a system which possesses a certain property or is operating at a certain state. Clearly, it is desirable to know how this property or state will be affected by a perturbation of some aspect of the system. Intuitively, for most systems we would like these disturbances to die out after a certain time, or at least to have only minimal effect on the property in question, and if this is the case, then we shall say it is *stable* with respect to perturbations of the class being considered.
Two approaches to the problem can be considered. The first one investigates the sensitivity of the systems state-space trajectories with respect to variations in the initial conditions. The method was developed in 1892 by A. M. Lyapunov, inspired by the use of the energy function in the analysis of classical mechanical systems, and we shall consider a generalization of this function.
Another concept of stability which is of great importance in control engineering, is the input-output stability (I/O stability). It is concerned with the sensitivity of the system's output signals with respect to "small" variations of the input signals.
We now formalize these notions.

Internal stability

This notion of stability for a linear system is strongly related to the properties of the transition matrix $\Phi(.,.)$ [93]. Indeed, consider the homogeneous equation

$$x_{k+1} = A_k x_k, \quad k \geq k_i, \tag{1.12}$$

whose solution is given by

$$x_k = \Phi(k, k_i) x_{k_i}, \quad k > k_i. \tag{1.13}$$

The system is said to be *asymptotically stable* if the solution x_k of (1.12) satisfies the condition

$$\|x_k\| \to 0 \quad \text{as} \quad k \to \infty \tag{1.14}$$

for any vector norm and for any initial state x_{k_i}.
Because of (1.13), (1.14) is equivalent to requiring that

$$\|\Phi(k, k_i)\| \to 0 \quad \text{as} \quad k \to \infty,$$

for any induced (or subordinate) matrix norm [80, 28]. The system is *uniformly exponentially stable* if constants α and ρ exist with $\alpha > 0$ and $\rho \in [0, 1)$ such that the solution x_k of (1.12) satisfies

$$\|x_k\| \leq \alpha \rho^{k-k_i} \|x_{k_i}\|, \quad \forall k \geq k_i$$

for any initial state x_{k_i}. And because of (1.13), this is also equivalent to requiring that

$$\|\Phi(k, k_i)\| \leq \alpha \rho^{k-k_i} \|x_{k_i}\|, \quad \forall k \geq k_i.$$

It follows from this equation that uniform exponential stability is preserved under a Lyapunov transformation (see §.1.2.3).
Uniform exponential stability is also equivalent to requiring that a positive integer τ and a positive constant $\gamma < 1$ exist such that

$$\|\Phi(k + \tau, k)\| < \gamma, \quad \forall k \geq k_1 \quad \text{for some} \quad k_1 \geq k_i. \tag{1.15}$$

Since

$$x_{k+\tau} = \Phi(k + \tau, k) x_k,$$

(1.15) implies that

$$\|x_{k+\tau}\| \leq \gamma \|x_k\| \quad \text{for} \quad k \geq k_i.$$

An important concept regarding internal stability is Lyapunov stability, which we outline below.

Lyapunov stability

Lyapunov stability analysis plays an important role in the stability analysis of control systems described by state equations.
There are two methods of stability analysis due to Lyapunov. The first method consists entirely of procedures in which the explicit forms of the solutions are used for the analysis. The second method, on the other hand, does not require the solution, in the sense that it is applicable to both linear and nonlinear systems, time-invariant or time-varying. This is also called the *direct method of Lyapunov*. It is based on a generalization of the fact that from the classical theory of mechanics, we know that a vibrating system is stable if its total energy is continually decreasing until an equilibrium state is reached. The idea is, however, more general than that of energy and is more widely applicable. We shall now present a stability theorem based on this method.

Definition 1.6.

- *A time-varying function $V(x,k)$ is said to be positive definite if there exists a positive definite function $V(x)$ s.t.*
  - $V(x,k) > V(x) \quad \forall k \geq k_0$,
  - $V(0,k) = 0 \quad \forall k \geq k_0$.
- *A time-varying function $V(x,k)$ is said to be negative definite if $-V(x,k)$ is positive definite.*

Theorem 1.7. *[93, 28, 80] Consider the discrete-time system*

$$x_{k+1} = A_k x_k. \tag{1.16}$$

Suppose there exists a continuous function $V(x,k)$ s.t.

- *$V(x,k)$ is positive definite,*
- *$\Delta V(x)$ is negative definite, where*

$$\Delta V(x,k) = V(x,k+1) - V(x,k),$$

then the equilibrium state at the origin is uniformly asymptotically stable and V is called a Lyapunov function. ■

Suppose we postulate a quadratic Lyapunov function

$$V(x,k) = x_k^* P_k x_k,$$

where P_k is Hermitian and positive definite for all k. Then, we easily obtain

$$\Delta V(x,k) = x_k^*(A_k^* P_{k+1} A_k - P_k)x_k.$$

Let

$$Q_k = -(A_k^* P_{k+1} A_k - P_k), \tag{1.17}$$

which must also be a Hermitian matrix. (1.17) is a Stein equation or a discrete-time Lyapunov equation (§.1.2.7). Its solvability (for a given $A_\bullet$ and $Q_\bullet$) relates directly to the (asymptotic) stability of the system (1.16).
In fact, a necessary and sufficient condition for uniform exponential stability is that a Hermitian positive definite matrix P_k exists with $c_1 I \leq P_k \leq c_2 I$ for some positive constants c_1 and c_2 so that

$$A_k^* P_{k+1} A_k - P_k \leq -c_3 I$$

for some positive constant c_3 [93]. These constants are time independent (i.e., independent of k).

I/O stability

The concept of I/O stability, roughly asserts that "any bounded input $u_\bullet$ produces a bounded output $y_\bullet$", more precisely iff

$$\exists\, \alpha < \infty \quad \text{s.t for all bounded} \quad u_\bullet, \quad \text{we have} \quad \|y_\bullet\|_\infty \leq \alpha \|u_\bullet\|_\infty. \tag{1.18}$$

The discrete-time system described by (1.5) is I/O stable iff the I/O map $\mathcal{H}$ is bounded. Hence, this is equivalent to say that the norm

$$\|\mathcal{H}\|_{\mathcal{H}_\infty}^2 \doteq \sup_{u_\bullet \neq 0} \frac{\|y_\bullet\|_2}{\|u_\bullet\|_2}$$

is bounded.

Theorem 1.8. *[28] The system is I/O stable iff*

$$\sup_k \{ \sum_{i=-\infty}^{k} \|H(k,i)\| \} \doteq \alpha < \infty$$

where the norm of $H(k,i)$ is the induced matrix norm

$$\|M\| \doteq \max_i \sum_j |m_{ij}|.$$

■

Using (1.4) this is equivalent to

$$\sup_{k\geq 0} \left\{ \sum_{i=0}^{k-1} \|C_k \Phi(k,i+1) B_i\| \right\} < \infty \tag{1.19}$$

(where the matrix norm used is arbitrary).

1.2.5 Controllability and observability

Controllability and observability belong to the fundamental notions of state-space theory. They are related to two problems concerning the state of a system :

- the influence of the input-functions on the state of the system.
  Question : Is it possible to transfer any state of the system to any other state by a suitable control function ?
  This problem involves the notion of *controllability*,
- the effect of the state on the output-functions of the system.
  Question : Is it possible to determine the initial state of the system by observing the output-function for a certain time (while the input-function is known)?
  This is related to the notions of *observability* for which we observe the output for a zero input.

In "The control handbook" [1] one can read the defining terms.
controllability : the existence of inputs that drive a system from any initial state to any desired state.
observability : The ability to compute the initial state x_{k_i} from knowledge of the output response y_k for $k \geq k_i$.

In the sequel we formalize these notions

Controllability

We denote $\mathcal{U}(k_i, k_1)$ the linear space of input sequences

$$U(k_i,k_1) = \left[u_{k_1-1}^T \; u_{k_1-2}^T \ldots u_{k_i}^T \right]^T,$$

and by $\mathcal{Y}(k_i, k_1)$ the linear space of output sequences

$$Y(k_i,k_1) = \left[y_{k_i}^T \ldots y_{k_1-2}^T \; y_{k_1-1}^T \right]^T.$$

Since controllability involves the relation between the input $u_\bullet$ and the state trajectory $x_\bullet$, we are interested in the pair $(A_\bullet, B_\bullet)$.
We start by defining the *controllability* (or *reachability*) *matrix* on any interval $[k_i, k_1]$ by

$$\mathcal{C}(k_i,k_1) \doteq \left[B_{k_1-1} \middle| \Phi(k_1, k_1-1) B_{k_1-2} \middle| \ldots \middle| \Phi(k_1, k_i+1) B_{k_i} \right].$$

Definition 1.9. *The pair $(A_\bullet, B_\bullet)$ is controllable on $[k_i, k_1]$ iff for any states x_{k_1} at time $k = k_1$ and x_{k_i} at time $k = k_i$, there exists a control sequence $U(k_i, k_1)$ that transfers the phase (x_{k_i}, k_i) to the phase (x_{k_1}, k_1), i.e.,*

$$x_{k_1} = \Phi(k_1, k_i)x_{k_i} + \underbrace{\sum_{i=k_i}^{k_1-1} \Phi(k_1, i+1)B_i u_i}_{\mathcal{C}(k_i,k_1)U(k_i,k_1)} \tag{1.20}$$

In this case, we say that the state x_{k_1} can be reached from x_{k_i}.

Remark 1.10. Sometimes we do not want to specify k_1 : then we talk about controllability at k_i. (i.e., iff for some $k > k_i$ the pair $(A_\bullet, B_\bullet)$ is controllable on $[k_i, k]$).

Definition 1.11. *The system (1.2) is controllable on $[k_i, k_1]$ if any state x_1 can be reached from any other state x_0 on $[k_i, k_1]$.*

Proposition 1.12. *The system (1.2), because it is linear, is controllable on $[k_i, k_1]$ iff every state x can be reached from the zero state, $x_{k_i} = 0$, at an instant k so that $k_i \le k \le k_1$.*

The states that can be reached from $x_{k_i} = 0$ at instant k ($k_i \le k \le k_1$) are

$$\{x \in \mathbb{C}^N \mid x = x_k = \mathcal{C}(k_i, k)U(k_i, k) = \sum_{i=k_i}^{k-1} \Phi(k, i+1)B_i u_i\},$$

which implies that

$$x \in span\{B_{k-1}, \Phi(k, k-1)B_{k-2}, \ldots, \Phi(k, k_i+1)B_{k_i}\}.$$

Hence, for any states $x_{k_i} = x_0$ and $x_{k_1} = x_1$, from (1.20) the input sequence $U(k_i, k_1)$ drives the system from x_0 to x_1 iff

$$rank(\mathcal{C}(k_i, k_1)) = N, \tag{1.21}$$

where N is the system order. If this is the case, (1.20) can be solved for $U(k_i, k_1)$, giving

$$U(k_i, k_1) = \mathcal{C}(k_i, k_1)^* \left[\mathcal{C}(k_i, k_1)\mathcal{C}(k_i, k_1)^*\right]^{-1} \left[x_1 - \Phi(k_1, k_i)x_0\right].$$

And so (1.21) is a necessary and sufficient condition for controllability over $[k_i, k_1]$.

Now let us define the *controllability map* on $[k_i, k_1]$

$$\begin{aligned} \mathcal{L}_c(k_i, k) \ &: \mathcal{U}(k_i, k) \to \mathbb{C}^N \\ &U(k_i, k) \to \mathcal{C}(k_i, k)U(k_i, k) = \sum_{i=k_i}^{k-1} \Phi(k, i+1)B_i u_i \end{aligned} \tag{1.22}$$

which maps the input sequence $U(k_i, k) \in \mathcal{U}(k_i, k)$ to the state $x_k \in \mathbb{C}^N$, with zero initial state, $x_{k_i} = 0$. Then we can define the adjoint operator

$$\mathcal{L}_c^*(k_i, k) \ : \ \mathbb{C}^N \ \to \ \mathcal{U}(k_i, k) \tag{1.23}$$

of $\mathcal{L}_c(k_i, k)$, which must satisfy,

$$\langle \mathcal{L}_c(k_i, k)u, x\rangle = \langle u, \mathcal{L}_c^*(k_i, k)x\rangle, \tag{1.24}$$

and hence satisfies :

$$(\mathcal{L}_c^*(k_i, k)x)_i = B_i^* \Phi(k, i+1)^* x.$$

The system is controllable iff Im $\mathcal{L}_c(k_i,k) = \mathbb{C}^N$, and hence iff $\mathcal{L}_c(k_i,k)\mathcal{L}_c^*(k_i,k)$ is positive definite. The controllability Gramian, $\mathcal{G}_c(k_i,k) \ : \ \mathbb{C}^N \ \rightarrow \ \mathbb{C}^N$, is defined as

$$\mathcal{G}_c(k_i,k) \doteq \sum_{i=k_i}^{k-1} \Phi(k,i+1)B_iB_i^*\Phi(k,i+1)^* = \mathcal{C}(k_i,k)\mathcal{C}(k_i,k)^*. \tag{1.25}$$

A first comment, is that (1.25) shows that $\mathcal{G}_c(k_i,k)$ is the sum of Hermitian positive semi-definite matrices and is therefore itself Hermitian positive semi-definite, whence $\forall z \in \mathbb{C}^N, \quad z^*\mathcal{G}_c(k_i,k)z \geq 0$. Note that

$$\mathcal{G}_c(k_i,k) = \mathcal{L}_c(k_i,k)\mathcal{L}_c(k_i,k)^*. \tag{1.26}$$

A second comment is that $\mathcal{G}_c(k_i,k)$ is the solution $X(k)$ of the forward matrix recursion equation called the *forward time-varying Stein equation.*

$$X(i+1) = A_iX(i)A_i^* + B_iB_i^*, \qquad i \geq k_i \tag{1.27}$$

with the initial condition $X(k_i) = 0$ (we refer to §.1.2.7 for more details on this equation).

Finally, we recall all controllability results in the following theorem :

Theorem 1.13. *[28] The following assertions are equivalent :*

1. *The pair* $(A_\bullet, B_\bullet)$ *is controllable on* $[k_i,k_1]$,
2. *Im* $\mathcal{L}_c(k_i,k_1) = \mathbb{C}^N$,
3. *rank* $\mathcal{C}(k_i,k_1) = N$,
4. det $\mathcal{G}_c(k_i,k_1) \neq 0$.

■

Observability

For any initial x_{k_i} at k_i and any input sequence $U(k_i,k_1) \in \mathcal{U}(k_i,k_1)$, the corresponding output sequence $Y(k_i,k_1) \in \mathcal{Y}(k_i,k_1)$ is given by

$$y_k = C_k\Phi(k,k_i)x_{k_i} + \underbrace{C_k \sum_{i=k_i}^{k-1} \Phi(k,i+1)B_iu_i}_{\mathcal{C}(k_i,k)U(k_i,k)}, \qquad \forall k \in [k_i,k_1]. \tag{1.28}$$

Thus for the study of the relation between the state x_{k_i} and the outputs $Y(k_i,k_1)$ we need only to consider the first term, i.e.,

$$y_k = C_k\Phi(k,k_i)x_{k_i}, \ \forall k \in [k_i,k_1], \quad \text{i.e.} \quad Y(k_i,k_1) = \mathcal{O}(k_i,k_1)x_{k_i} \tag{1.29}$$

which is known if the pair of matrix-sequences $C_\bullet$ and $A_\bullet$ are known, where

$$\mathcal{O}(k_i,k_1) \doteq \begin{bmatrix} C_{k_i} \\ C_{k_i+1}\Phi(k_i+1,k_i) \\ \vdots \\ C_{k_1}\Phi(k_1,k_i) \end{bmatrix} \tag{1.30}$$

is the *observability matrix* on $[k_i,k_1]$.

Definition 1.14. *The pair* $(A_\bullet, C_\bullet)$ *is observable on* $[k_i,k_1]$ *iff for all input sequences* $U(k_i,k_1)$ *and for all corresponding output sequences* $Y(k_i,k_1)$*, the state* x_{k_i} *at* k_i *is uniquely determined.*

Proposition 1.15. *[28] The system* (1.2)*, because it is linear, is observable on* $[k_i, k_1]$ *iff the zero state is the only state which results in the zero output,* $y_{[k_i,k_1]} \equiv 0$*, with zero input,* $u_{[k_i,k_1]} \equiv 0$*.* ■

According to (1.28), the states which result in the zero output, $y_k \equiv 0$, with zero input, $u_k \equiv 0$, are

$$\{z \in \mathbb{C}^N \mid C_k \Phi(k, k_i) z \equiv 0\},$$

which implies

$$z \in \text{Ker } \mathcal{O}(k_i, k_1).$$

The system is observable iff

$$\text{Ker } \mathcal{O}(k_i, k_1) = \{0\},$$

and iff

$$rank\ \mathcal{O}(k_i, k_1) = N.$$

(1.29) suggests to define the *observability map* on $[k_i, k_1]$ as

$$\begin{aligned} \mathcal{L}_o(k, k_1) \ &: \mathbb{C}^N \to \mathcal{Y}(k, k_1) \\ x \ &\to \mathcal{O}(k, k_1) x. \end{aligned} \tag{1.31}$$

This maps x to the output y_k resulting from the initial state $x_{k_i} = \Phi(k_i, k_1)x$ and zero input. The system (1.2) is observable iff $\text{Ker}\mathcal{L}_o(k, k_1) = \{0\}$, and hence iff $\mathcal{L}_o^*(k, k_1)\mathcal{L}_o(k, k_1)$ is positive definite.
The observability Gramian, $\mathcal{G}_o(k, k_1) : \mathbb{C}^N \to \mathbb{C}^N$, is defined as

$$\mathcal{G}_o(k, k_1) \doteq \sum_{i=k}^{k_1} \Phi^*(k, i) C_i^* C_i \Phi(k, i) = \mathcal{L}_o^*(k, k_1)\mathcal{L}_o(k, k_1). \tag{1.32}$$

This Gramian is also the solution $X(k)$ of the backward time-varying Stein equation

$$X(i) = A_i^* X(i+1) A_i + C_i^* C_i, \qquad i \leq k_1, \tag{1.33}$$

with the initial condition $X(k_1 + 1) = 0$.

Finally, we resume all observability results in the following theorem :

Theorem 1.16. *[28] The following assertions are equivalent :*

1. *The pair* $(A_\bullet, C_\bullet)$ *is observable on* $[k_i, k_1]$*,*
2. *Ker* $\mathcal{L}_o(k_i, k_1) = \{0\}$*,*
3. *rank* $\mathcal{O}(k_i, k_1) = N$*,*
4. det $\mathcal{G}_o(k_i, k_1) \neq 0$*.*

■

1.2.6 Energy storage

Given a system with initial state x_{k_i}, then we can describe the energy of the future output, and the minimal energy needed in the past to arrive at this initial state in terms of the Gramians. Let us make this more precise. According to (1.26) we have that on $[k_i, k_1]$, for any state x

$$\langle x, G_c(k_i, k) x \rangle = \|\mathcal{L}_c^*(k_i, k) x\|^2.$$

Now, if we define the generalized inverse of $\mathcal{L}_c(k_i, k)$,

$$\mathcal{L}_c^\dagger(k_i,k) \; : \; \mathbb{C}^N \; \rightarrow \; \mathcal{U}(k_i,k),$$

by

$$\mathcal{L}_c^\dagger(k_i,k) = \mathcal{L}_c^*(k_i,k)(\mathcal{L}_c(k_i,k)\mathcal{L}_c^*(k_i,k))^{-1} = \mathcal{L}_c^*(k_i,k)\mathcal{G}_c(k_i,k)^{-1},$$

then $u = \mathcal{L}_c^\dagger(k_i,k)x$ is the unique solution to $\mathcal{L}_c(k_i,k)u = x$ of the smallest norm :

$$\|\mathcal{L}_c^\dagger(k_i,k)x\| < \|u\|, \quad \forall u \; : \mathcal{L}_c(k_i,k)u = x, u \neq \mathcal{L}_c^\dagger(k_i,k)x.$$

Its norm is

$$\|\mathcal{L}_c^\dagger(k_i,k)x\|^2 = \|\mathcal{L}_c^*(k_i,k)\mathcal{G}_c^{-1}(k_i,k)x\|^2 = \langle x, \mathcal{G}_c^{-1}(k_i,k)x\rangle = x^*\mathcal{G}_c^{-1}(k_i,k)x.$$

Thus, given any two states x_0 and x, the unique input sequence that minimizes $\|u\|$ among all inputs which take x_0 to x in $[k_i,k]$ is given by

$$u_i = B_i^*\Phi^*(k,i+1)\mathcal{G}_c^{-1}(k_i,k)(x - \Phi(i,k_i)x_0), \quad i = 1:k.$$

Similarly, given a sequence of output $y \in \mathcal{Y}(k,k_1)$, the initial state x_0 is retrieved by

$$x_0 = \mathcal{G}_o(k,k_1)^{-1}\mathcal{L}_o^*(k,k_1)y.$$

Furthermore, we have

$$\|y\|^2 = x_0^*\mathcal{G}_o(k,k_1)x_0.$$

Thus, x_0 is the initial state that results in y as an output and maximizes $\|y\|$.

1.2.7 Stein, Lyapunov and Sylvester equations

Certain matrix equations arise naturally in linear control and system theory. Among those frequently encountered in the analysis and design of discrete-time linear systems are the generalized Stein or discrete-time Lyapunov equation

$$AXA^* - EXE^* + Q = 0, \tag{1.34}$$

and the generalized Sylvester equation

$$AXF - EXG + Q = 0. \tag{1.35}$$

By choosing $E = I$ in (1.34), and $E = I$ and $G = I$ in (1.35), we get the standard Stein or discrete-time Lyapunov equation and the standard Sylvester equation.

There is an extensive literature on these equations, but few pay attention to numerical aspects such as numerical stability, conditioning, machine implementation, and the like. We refer to the following selection of fundamental papers [15, 44, 58, 62, 68, 79, 95, 76, 77] which all present reliable algorithms.
Several methods for solving these equations have been developed. The most appealing algorithm in terms of efficiency, accuracy, reliability, availability, and ease of use appears to be that of Bartels and Stewart [15], and its variant proposed by Hammarling [68].
The basic idea is to reduce A to quasi-upper-triangular form (or real Schur form) and perform a back substitution for the elements of X. Hammarling instead solves directly for the *Cholesky factor*[7] Y of X : $Y^*Y = X$, and Y is upper triangular. In many applications, for example model reduction, this is useful and cheaper since only the Cholesky factor is required.

[7] **Cholesky factorization :** Expressing a Hermitian matrix M as a product of a lower or upper triangular matrix L and its conjugate transpose L^*, that is, $M = LL^*$.

The two equations (1.34) and (1.35) can also be written as linear systems using the Kronecker product matrix representation, and we obtain respectively :

$$(\bar{A} \otimes A - \bar{E} \otimes E)vec(X) = vec(Q), \tag{1.36}$$

and

$$(F^T \otimes A - G^T \otimes E)vec(X) = vec(Q). \tag{1.37}$$

Equation (1.36) has a unique solution iff the generalized eigenvalues λ_i of $A - \lambda E$ (i.e., $\lambda_i(A, E)$) satisfy

$$\lambda_i \bar{\lambda}_j \neq 1 \quad \forall i, j \quad \text{(with the convention that} \quad 0.\infty = 1).$$

If Q is (semi)definite and $|\lambda_i(A, E)| < 1$ for all i, then a unique (semi)definite solution exists [95]. If Q is symmetric, then X is symmetric as well.
If E is invertible, we obtain the solution

$$X = \sum_{i=0}^{\infty} (E^{-1}A)^i E^{-1} Q E^{-*} (A^* E^{-*})^i.$$

Furthermore, when (E, A) is stable and E invertible, the solutions of this equation are also related to the controllability and observability Gramians $\mathcal{G}_c$ and $\mathcal{G}_o$ corresponding to the pairs $(E^{-1}A, E^{-1}B)$ and (AE^{-1}, CE^{-1}) respectively.
These Gramians are respectively the solutions of the equations :

$$AG_cA^* - EG_cE^* + BB^* = 0, \tag{1.38}$$

and

$$A^*G_oA - E^*G_oE + C^*C = 0. \tag{1.39}$$

They are easily checked to be equal to :

$$G_c = \sum_{i=0}^{\infty} (E^{-1}A)^i E^{-1} B B^* E^{-*} (A^* E^{-*})^i, \tag{1.40}$$

and

$$G_o = \sum_{i=0}^{\infty} (E^{-*}A^*)^i E^{-*} C^* C E^{-1} (AE^{-1})^i. \tag{1.41}$$

By Parseval's theorem these are also equal to :

$$G_c = \frac{1}{2\pi} \int_0^{2\pi} (e^{j\omega}E - A)^{-1} B B^* (e^{-j\omega}E - A)^{-*}, \tag{1.42}$$

and

$$G_o = \frac{1}{2\pi} \int_0^{2\pi} (e^{-j\omega}E - A)^{-*} C^* C (e^{j\omega}E - A)^{-1}. \tag{1.43}$$

Then the relationship between theses solutions and the Gramians is given by :

$$\mathcal{G}_c = G_c, \quad \text{and} \quad \mathcal{G}_o = E^*G_oE.$$

In fact, G_c and G_o do not correspond to the Gramians of the "equivalent" explicit systems

$$\{E^{-1}A, E^{-1}B, C\} \quad \text{or} \quad \{AE^{-1}, B, CE^{-1}\},$$

but to a mixture of both.

Equation (1.37) has a unique solution if $A - \lambda E$ and $G - \lambda F$ are nonsingular and have disjoint spectra, i.e., $\lambda_i/\mu_j \neq 1$, $\forall i, j$ where λ_i denote the eigenvalues of $A - \lambda E$ and μ_i denote the eigenvalues of $G - \lambda F$. If E and G are invertible, we obtain the solution

$$X = \sum_{i=0}^{\infty} (E^{-1}A)^i E^{-1} Q G^{-1} (F G^{-1})^i.$$

For the time-varying case, the analogs of (1.34) are the *Stein recurrences* :

$$A_k X_k A_k^* - E_k X_{k+1} E_k^* + Q_k = 0, \tag{1.44}$$

and

$$A_k^* X_{k+1} A_k - E_k^* X_k E_k + Q_k = 0. \tag{1.45}$$

The first equation is called the *forward Stein recurrence* and the second equation the *backward Stein recurrence.*
Now, suppose that all E_i are nonsingular for all i, and let us consider a finite window $[k_i, k_f]$. When the generalized transition matrices $\Phi_{LG}(k, k_i)$ and $\Phi_{RG}(k, k_i)$ defined by

$$\Phi_{LG}(i, j) \doteq E_{i-1}^{-1} A_{i-1} E_{i-2}^{-1} A_{i-2} \dots E_j^{-1} A_j \quad (i \geq j),$$

and

$$\Phi_{RG}(i, j) \doteq A_{i-1} E_{i-1}^{-1} A_{i-2} E_{i-2}^{-1} \dots A_j E_j^{-1} \quad (i \geq j),$$

are stable for all $i, j \in (k_i, k_f]$, the solutions $G_c(k)$ and $G_c(k)$ of these equations corresponding to the initial conditions

$$\mathcal{G}_c(k_i) = 0, \quad \mathcal{G}_o(k_f + 1) = 0.$$

are also related to the controllability and observability Gramians $\mathcal{G}_c(k)$ and $\mathcal{G}_o(k)$ of the time-varying system $\{E_\bullet, A_\bullet, B_\bullet, C_\bullet\}$ as follows :

$$\mathcal{G}_c(k) = G_c(k) = \sum_{i=k_i}^{k-1} \Phi_{LG}(k, i+1) B_i B_i^* \Phi_{LG}^*(k, i+1),$$

and

$$\mathcal{G}_o(k) = E_k^* G_o(k) E_k = E_k^* \sum_{i=k}^{k_f} \Phi_{RG}^*(i, k) C_i^* C_i \Phi_{RG}(i, k) E_k.$$

1.2.8 System characteristics and norms

In this section we briefly summarize some characteristic system parameters, we introduce the frequency response and define system norms. Parameters that are characteristic of a system are realization independent. Here we introduce Hankel singular values (HSV), frequency response, Markov parameters, moments and system norms.

Hankel singular values

For a stable linear time-invariant system with invertible matrix

$$\mathcal{S} \quad \begin{cases} Ex_{k+1} = Ax_k + Bu_k \\ y_k = Cx_k, \end{cases}$$

we can define using the Gramians $\mathcal{G}_c$ and $\mathcal{G}_o$[8], very useful invariant parameters called the *Hankel singular values* (HSV) by :

$$\sigma_i(\mathcal{S}) \doteq (\lambda_i(\mathcal{G}_c\mathcal{G}_o))^{1/2}.$$

These values are independent of the realization (E, A, B, C), and are also the singular values of the Hankel operator $\mathcal{H}$.
It is important to point out here that for linear time-varying systems we can still define the eigenvalues of the product of the Gramians :

$$\sigma_i(k) \doteq (\lambda_i(\mathcal{G}_c(k)\mathcal{G}_o(k)))^{1/2},$$

and these will be positive real if the system is controllable and observable over the considered finite time window $[k_i, k_f]$ (§ 1.2.2) (G_c and G_o are given by (1.40) and (1.41)). These $\sigma_i(k)$ are the time-varying Hankel singular values corresponding to the input-output map on this window (1.7).

Frequency response

The evaluation of the transfer function $T_f(z) = C(zE - A)^{-1}B$ for $z = e^{j\omega}$, $\omega \in [-\pi, \pi]$ defines the *frequency response* of the system. The maximum singular value frequency response function is defined as

$$\sigma(T_f(.)) = \lambda_{max}(T_f(e^{j\omega})T_f^*(e^{j\omega}))^{\frac{1}{2}}.$$

Markov parameters and moments

A transfer function matrix $T_f(z) = C(zE - A)^{-1}B$ can be expanded into a power series in z^{-1} :

$$T_f(z) = \sum_{i=1}^{\infty} H_i z^{-i},$$

$$H_i = C(E^{-1}A)^{i-1}E^{-1}B.$$

The parameters H_i are called *Markov parameters* of the system $\mathcal{S}$. This expansion in fact is made around the point at infinity. We can also expand the transfer function around a finite point in the complex plane σ, and we obtain

$$T_f(z) = \sum_{j=1}^{\infty} \eta_j(\sigma)(z - \sigma)^j,$$

$$\eta_j(\sigma) = C((A - \sigma E)^{-1}E)^j(\sigma E - A)^{-1}B.$$

The parameters $\eta_j(\sigma)$ are called *moments* of the transfer function $T_f(z)$ at σ, and these are independent of the chosen realization.

[8] In this case we have $\mathcal{G}_c = G_c$ and $\mathcal{G}_o = E^*G_oE$, where G_c and G_o are respectively the solutions of

$$AG_cA^* - EG_cE^* + BB^* = 0, \quad \text{and} \quad A^*G_oA - E^*G_oE + C^*C = 0.$$

(see §.1.2.7 for more details)

System norms

Finally, we review definitions and some properties of three system norms $\mathcal{H}_2$, $\mathcal{H}_\infty$, and Hankel. This is crucial in measuring system responses.

The $\mathcal{H}_2$ norm

The $\mathcal{H}_2$ norm of a system $\mathcal{S}$ (with the transfer function T_f) is defined as

$$\|\mathcal{S}\|^2_{\mathcal{H}_2} = \frac{1}{2\pi}\sum_{-\infty}^{+\infty} Trace(T_f(e^{j\omega})T_f^*(e^{j\omega})),$$

where $T_f^*(e^{j\omega}) = T_f^T(e^{-j\omega})$.
With $H_i = C(E^{-1}A)^{i-1}E^{-1}B$ the impulse response of $\mathcal{S}$ we also have due to Parseval's relation [137] :

$$\|\mathcal{S}\|^2_{\mathcal{H}_2} = \sum_{1}^{\infty} Trace(H_iH_i^*).$$

Furthermore, a convenient way to determine its numerical value is to use the following formulas

$$\|\mathcal{S}\|^2_{\mathcal{H}_2} = Trace(CG_cC^*) = Trace(B^*G_oB).$$

This property can easily be proven as follows :

$$\begin{aligned}
\|\mathcal{S}\|^2_{\mathcal{H}_2} &= \sum_{1}^{\infty} Trace(H_iH_i^*) \\
&= \sum_{1}^{\infty} Trace(C(E^{-1}A)^{i-1}E^{-1}BB^*E^{-*}((E^{-1}A)^*)^{i-1}C^*) \\
&= \sum_{0}^{\infty} Trace(C(E^{-1}A)^{i}E^{-1}BB^*E^{-*}((E^{-1}A)^*)^{i}C^*) \\
&= Trace\left(C\underbrace{\left(\sum_{0}^{\infty}(E^{-1}A)^iE^{-1}BB^*E^{-*}((E^{-1}A)^*)^i\right)}_{G_c}C^*\right) \\
&= Trace\left(CG_cC^*\right)
\end{aligned}$$

where G_c is the solution of (1.40). Similarly, we have

$$\begin{aligned}
\|\mathcal{S}\|^2_{\mathcal{H}_2} &= \sum_{0}^{\infty} Trace(C(E^{-1}A)^{i}E^{-1}BB^*E^{-*}((E^{-1}A)^*)^{i}C^*) \\
&= \sum_{0}^{\infty} Trace(B^*E^{-*}((E^{-1}A)^*)^{i}C^*C(E^{-1}A)^{i}E^{-1}B) \\
&= Trace\left(B^*E^{-*}\underbrace{\left(\sum_{0}^{\infty}((E^{-1}A)^*)^iC^*C(E^{-1}A)^i\right)}_{X}E^{-1}B\right)
\end{aligned}$$

where X is the solution of

$$A^*E^{-*}XE^{-1}A - X + C^*C = 0.$$

We have $G_o = E^{-*}XE^{-1}$, where G_o is solution of (1.41). So,

$$\|\mathcal{S}\|^2_{\mathcal{H}_2} = Trace\left(B^*E^{-*}XE^{-1}B\right) = Trace\left(B^*G_oB\right).$$

The $\mathcal{H}_\infty$ norm

The $\mathcal{H}_\infty$ norm of a stable system $\mathcal{S}$ is defined as

$$\|\mathcal{S}\|_{\mathcal{H}_\infty} = \max_\omega \sigma_{max}(T_f(e^{j\omega})).$$

Interpretation of $\mathcal{H}_\infty$ and $\mathcal{H}_2$ norm

A very nice interpretation of these two norms can be formulated for SISO systems.
Indeed, the $\mathcal{H}_\infty$ norm is the peak of the transfer function magnitude (in terms of its singular values), and the $\mathcal{H}_2$ norm is the *expectation* of the output energy represented by the surface delimited by this transfer function magnitude as it is shown in Figure 1.3.

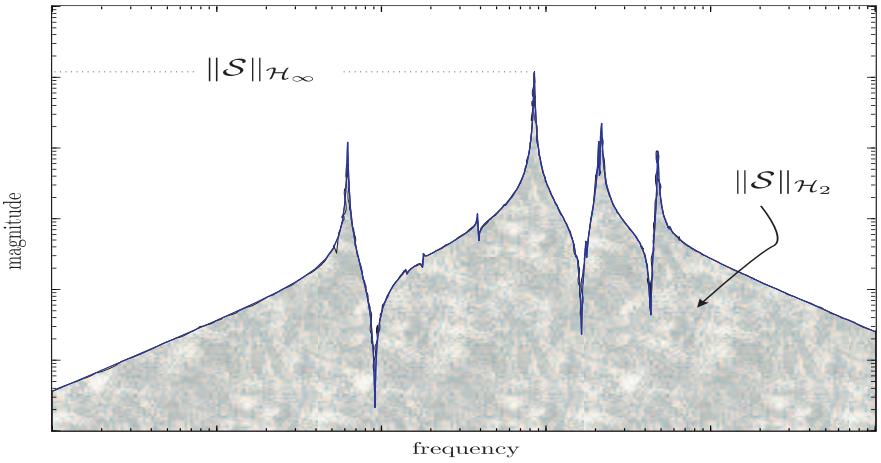


Fig. 1.3. Interpretation of $\mathcal{H}_\infty$ and $\mathcal{H}_2$ norm for a SISO system.

The Hankel-norm

The Hankel-norm is the largest Hankel singular value of the system

$$\|\mathcal{S}\|_H = \sigma_{max}(T_f(z)) = \sqrt{\lambda_{\max}(\mathcal{G}_c\mathcal{G}_o)}.$$

It is a measure of the effect of its past inputs on its future outputs, or the amount of energy stored in the system, and subsequently retrieved from it.

1.2.9 C/D and D/C conversions

In this text we consider mainly linear discrete-time systems, but actually this is not constraining for our results as they can be easily extended to continuous-time systems as well using conversion formulas between these two cases.
Actually, if one has to analyze, design, control or simulate a real world process, one often uses a discretized model in order to implement it on computer. The system can then be described by linear difference equations which are better suited for computation than the differential equations. So we consider that for numerical, algorithmic and computational point of view, discrete-time systems are more interesting.
But, discrete- and continuous-time systems are typically related by many transformations and conversions. It is well known that we can obtain a difference equation by approximating the derivatives with, for example, a forward or a backward difference (or Euler's method). Also *sampling* the signals can be made in different ways, for example, zero- or first-order hold [10].
In general, all results in continuous- and discrete-time systems are very similar, except that discrete transition matrices, summations, z-transform, $e^{j\omega}$, ... are replaced by continuous transition matrices, integrations, s-transform, $j\omega$, ..., respectively.

For the time-invariant case, a transformation between discrete- and continuous-time systems is the *Bilinear transformation* or *Tustin's approximation* given by $s = \frac{1}{\zeta}\frac{z-1}{z+1}$ where ζ is a design parameter called the *Shift parameter* and corresponds to the sampling time h. This transformation maps the unit disc to the left half plane and vice-versa (1.4).

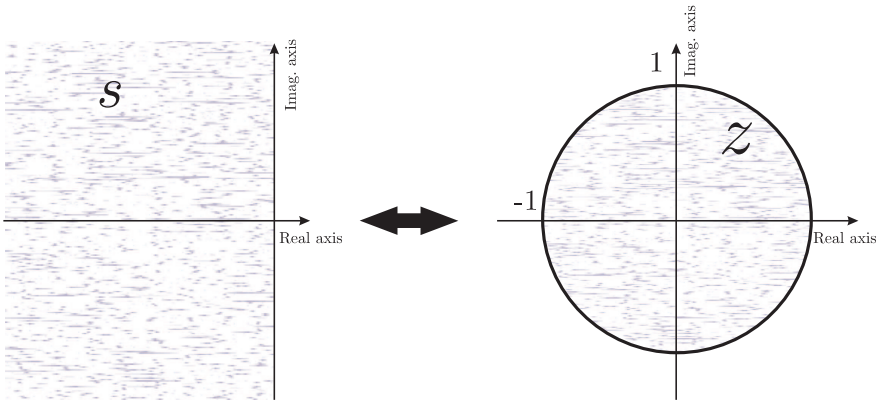


Fig. 1.4. Mapping the s-plane to the z-plane for the bilinear transformation.

Unlike Euler's method, this transformation has the considerable advantage that it conserves stability (or instability) of the original system in the transformed system [10]. The matrices of the state space descriptions are related as given in the Table 1.2 (subscript c to emphasize continuous-time).

discrete-time		continuous-time
A, B, C, D	$s = \frac{1}{\zeta}\frac{z-1}{z+1}$	$A^c = \frac{1}{\zeta}(A+I)^{-1}(A-I)$ $B^c = \frac{\sqrt{2\zeta}}{\zeta}(A+I)^{-1}B$ $C^c = \frac{\sqrt{2\zeta}}{\zeta}C(A+I)^{-1}$ $D^c = D - C(A+I)^{-1}B$
$A = (I-\zeta A^c)^{-1}(I+\zeta A^c)$ $B = \sqrt{2\zeta}(I-\zeta A^c)^{-1}B^c$ $C = \sqrt{2\zeta}C^c(I-\zeta A^c)^{-1}$ $D = D^c + C^c(I-\zeta A^c)^{-1}B_c$	$z = \frac{1+\zeta s}{1-\zeta s}$	A^c, B^c, C^c, D^c

Table 1.2. Transformation between continuous- and discrete-time systems and vice-versa for time-invariant case.

Remark 1.17. The transformation above preserves the Gramians and the infinity norm [5].

For the time-varying case, this bilinear transformation can be generalized as well. And we obtain for example a discrete-time system from a continuous-time systems with the correspondences :

$$\begin{aligned} A_k &= (I - hA^c(kh))^{-1}(I + hA^c(kh)), \\ B_k &= \sqrt{2h}(I - hA^c(kh))^{-1}B^c(kh), \\ C_k &= \sqrt{2h}C^c(kh)(I - hA^c(kh))^{-1}. \end{aligned}$$

We will see later what these correspondences will imply for our numerical schemes.

Chapter 2

Model reduction for linear systems

In the last twenty years, model reduction of large scale dynamical systems has become very popular. The idea is to construct a "simple" lower order model that approximates well the behaviour of a "complex" larger dynamical model. A complex system is essentially a mathematical model which describes a real world physical process. This mathematical model is often characterized by partial differential equations (PDEs) or ordinary differential equations (ODEs). Since improved accuracy (using e.g. a very fine discretization) leads to models of high complexity, this may become prohibitive for certain computations (control, optimization, ...). Therefore it is essential to design models of reduced complexity without sacrificing too much accuracy.

There are many ways to perform model reduction, but we can distinguish roughly three approaches for generating reduced order models of linear, time-invariant systems [6, 5].
A first class is a norm minimization which tries to approximate a model $\mathcal{S}$ by a lower-order model $\mathcal{S}_n$ so that input signals entering both $\mathcal{S}$ and $\mathcal{S}_n$ lead to a minimal difference at the outputs of $\mathcal{S}$ and $\mathcal{S}_n$. These methods are often referred to as *minimum distance methods*. The basic problem is [5] :

$$\min_{\mathcal{S}_n} \|\mathcal{S} - \mathcal{S}_n\|_p,$$

where p ($p = 2, \infty$) specifies the system norm. For example, for $p = \infty$ the underlying problem can be interpreted as an optimal attenuation problem : we search for $G_n(z)$ that achieves the lowest l_2^p-norm of the output of $G(z) - G_n(z)$ for any input with unit l_2^m-norm.
This problem has only a closed form solution for the optimal Hankel-norm approximation [3, 63, 92], and not for the more important optimal $\mathcal{H}_\infty$- and $\mathcal{H}_2$-norm approximation. Fortunately, the optimal Hankel-norm approximation yields a bounded distance measured in $\mathcal{H}_\infty$-norm [63, 50].

A second class of methods in model reduction is the reduction based on parameter matching. The idea is that a relatively small number of system parameters can be identified that are crucial in the description of the dynamical behaviour. By using model reduction schemes that preserve these parameters, an adequate lower-order description can be obtained. System parameters that can be matched are Markov parameters, and moments [43] to name a few. It is also possible to make an interpolation through a selected number of frequency response points. The well known methods in this class are : Padé [53, 51], Arnoldi [49], Lanczos [65, 54], ADI [22, 85, 88] and Krylov based methods [55, 64, 71, 74, 75, 81, 102, 111]. Another method of this class is the *q-Markov covariance equivalent realizations* (q-Markov COVERs) [4, 86, 87, 109, 110]. It preserves the first q-Markov parameters and the first q-output covariance (Markov parameters of the causal part of the power spectrum).

The third class of model reduction methods is characterized by the principle of projection of dynamics. Balanced Truncation is the best known method of this class [50, 91, 99, 105].
Recently, it has been shown in [129, 57, 56] that projection is universal, at least for SISO systems and under certain constraints for the MIMO case as well. Both norm minimizing methods and parameter matching methods can thus be viewed as projection based methods.
In [72] it has been even shown that a method of projection is necessary to obtain an optimal solution of the $\mathcal{H}_2$-norm minimization problem.

Balanced Truncation is widely used in practice mainly for three reasons. Firstly, for a reasonable size system order say, $N \leq 100$, it gives a satisfactory approximation in the majority of cases without having to solve a complicated minimization problem or having to choose a set of essential system parameters first. Secondly, this approximation can be obtained at relatively reasonable computational cost. Thirdly an a priori upper bound for errors between original plants and reduced-order models exists for the $\mathcal{H}_\infty$-norm, the preferred measure of approximation accuracy in engineering.

So far we have not addressed the problem of choosing the order of the approximation. The purpose of the model determines the "acceptable" order reduction in an implicit way; an explicit criterion for acceptable order reduction is hard to give, as we need to analyze the dynamics involved in order to obtain some sort of dynamics ranking. For systems with $N \leq 100$, this analysis can be done at a reasonable cost, including the problem of finding an appropriate value of n. But for large-scale models this pretreatment is prohibitive.

For large scale problems one has to use iterative methods to find an adequate approximation. In this respect, ideas based on balanced reduction methods are interesting since they offer the possibility to perform order selection during the computation of the projection spaces and not in advance.

However, a serious drawback of Balanced Truncation (and all direct methods in general) is that it ignores sparsity of the system if there is, and that it is not very easy to parallelize (note however the recent work of V.Merhmann and al. who try to parallelize some traditional model reduction methods). Its use is therefore limited if large, sparse systems have to be reduced. If one wants to exploit this sparsity for computational efficiency, we have to consider iterative methods which are very suitable for this and are often easy to parallelize as well.

For the time-varying case, up to now only a time-varying version of Balanced Truncation exists. The main idea is to apply at each time step a Balanced Truncation of the Gramians evaluated at each time step [73, 106, 107, 48]. This approach has the major drawback of being very time consuming.

This chapter gives an overview of the most used model reduction methods. Section 2.1 presents the basic definitions and notation of the projection of dynamics approach. The projection based method of Balanced Truncation and the optimal Hankel norm reduction method are discussed in Section 2.2. In Section 2.3 the matching of system parameters based on a projection formulation is worked out.

2.1 Projection of dynamics

Before proceeding with projection of dynamics, we describe first the generalized state space approach in order to consider a more general framework, which can include projection methods whose left and right projection matrices are not bi-orthogonal. We will however assume E invertible.

Definition 2.1 (Partitioning and truncation). *Consider the generalized linear system of order N*

$$\mathcal{S} \quad \begin{cases} Ex_{k+1} = Ax_k + Bu_k, \\ y_k = Cx_k, \end{cases}$$

and consider the following partition of the system matrices :

$$E = \begin{bmatrix} E_{11} & E_{12} \\ E_{21} & E_{22} \end{bmatrix}, A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, B = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}, C = \begin{bmatrix} C_1 & C_2 \end{bmatrix}, \quad \textit{where} \quad n < N,$$

then the system

$$\mathcal{S}_n \quad \begin{cases} E_{11}\check{x}_{k+1} = A_{11}\check{x}_k + B_1 u_k, \\ \check{y}_k = C_1 \check{x}_k, \end{cases}$$

is an n^{th}-order truncation of S. The truncation is obtained using the projection

$$\pi = \begin{bmatrix} I_n \\ 0 \end{bmatrix} \begin{bmatrix} I_n & 0 \end{bmatrix}.$$

The combination of applying a similarity transformation and a subsequent truncation is often referred to as projection of dynamics.

Definition 2.2. *Let $\pi_l, \pi_r \in \mathbb{C}^{N\times n}$ satisfy $\pi_l^*\pi_r = I_n$. Projection of dynamics of S is defined as*

$$\mathcal{S}_n \quad \begin{cases} \pi_l^* E \pi_r \hat{x}_{k+1} = \pi_l^* A \pi_r \hat{x}_k + \pi_l^* B u_k, \\ \hat{y}_k \qquad\quad = C\pi_r \hat{x}_k. \end{cases}$$

Projection of dynamics is also known as Transform and Truncate, and thus transformation by T and truncation by π are merged in the projection pair (π_l, π_r) as follows :

$$\pi_r = T \begin{bmatrix} I_n \\ 0 \end{bmatrix}, \quad \pi_l^* = \begin{bmatrix} I_n & 0 \end{bmatrix} T^{-1} \quad \text{and} \quad \pi = \pi_r \pi_l^*.$$

It can be verified easily that this definition satisfies $\pi_l^*\pi_r = I_n$ and hence $\pi^2 = \pi$.

A projection method is in fact a choice of two subspaces $\mathcal{S}_r, \mathcal{S}_l \subset \mathbb{C}^N$ of dimensions n, so that $\hat{x}. \in \mathcal{S}_r$ and the residual is orthogonal to $\mathcal{S}_l$. In fact, the columns of π_r and π_l form bases for $\mathcal{S}_r$ and $\mathcal{S}_l$, respectively,

$$\mathrm{Im}(\pi_r) = \mathcal{S}_r, \quad \mathrm{Im}(\pi_l) = \mathcal{S}_l.$$

If $\mathcal{S}_l = \mathcal{S}_r$, the projection is orthogonal, otherwise it is oblique. The subscripts r and l refer to right and left, respectively.
Actually the choice of basis of $\mathcal{S}_r$ and $\mathcal{S}_l$ is not important. If we take any two other base of these subspaces, e.g. $\bar{\pi}_r$ and $\bar{\pi}_l$, then there exist two invertible matrices $X, Y \in \mathbb{C}^{n\times n}$ such that

$$\bar{\pi}_r = \pi_r X, \quad \bar{\pi}_l = \pi_l Y.$$

For these two projector matrices we have :

$$\begin{aligned} \bar{T}_{f_n}(z) &= C\bar{\pi}_r(z\bar{\pi}_l^* E\bar{\pi}_r - \bar{\pi}_l^* A\bar{\pi}_r)^{-1}\bar{\pi}_l^* B \\ &= C\pi_r X(zY^*\pi_l^* E\pi_r X - Y^*\pi_l^* A\pi_r X)^{-1} Y^*\pi_l^* B \\ &= C\pi_r X\big(Y^*(z\pi_l^* E\pi_r - \pi_l^* A\pi_r)X\big)^{-1} Y^*\pi_l^* B \\ &= C\pi_r(z\pi_l^* E\pi_r - \pi_l^* A\pi_r)^{-1}\pi_l^* B \\ &= T_{f_n}(z) \end{aligned}$$

A special case of projection of dynamics is Balanced Truncation.

2.2 Balanced Truncation

The method of Balanced Truncation of linear systems is well established for model reduction. It is based on a balanced realization of the system. This realization has some nice sensitivity properties with respect to poles, zeros, truncation errors in digital filter implementations, and so on ([91, 137]). It is therefore recommended whenever the choice of a realization is not specified by the user.

For linear time-invariant systems, the approach requires standard matrix computations, and has been successfully used in control systems design. The main idea is to rewrite the system $\mathcal{S}$,

$$\mathcal{S} \quad \begin{cases} Ex_{k+1} = Ax_k + Bu_k, \\ y_k \quad = Cx_k, \end{cases}$$

which we suppose stable, controllable and observable[1], using a transformation T in the so-called balanced coordinate system and then use a truncation to obtain the reduced model. In this coordinate system one has [61] :

$$T\mathcal{G}_c T^* = T^{-*}\mathcal{G}_o T^{-1} = \Sigma = diag\{\sigma_1, \sigma_2, \ldots, \sigma_N\}$$

where the σ_i are the Hankel singular values of $\mathcal{S}$ and $\mathcal{G}_c$ and $\mathcal{G}_o$ are the controllability and observability Gramians of $\mathcal{S}$ [137].

A natural question now arises : what is the use of balancing, i.e., diagonalizing $\mathcal{G}_c$ and $\mathcal{G}_o$? This can be explained using energy functions. Given a stable linear system S, it then follows from (1.2.6) that for any state x :

$$\epsilon_c(x) = (x^*\mathcal{G}_c^{-1}x)^{\frac{1}{2}}, \quad \epsilon_o(x) = (x^*\mathcal{G}_o x)^{\frac{1}{2}}$$

are respectively the smallest amount of energy needed to steer the system from 0 to x, and the largest amount of energy obtained by observing the output of the free system with the initial condition x.
If we define the energy storage efficiency by

$$\epsilon(x_0) = \frac{x_0^*\mathcal{G}_o x_0}{x_0^*\mathcal{G}_c^{-1}x_0}, \tag{2.1}$$

then the maximization of $\epsilon(x_0)$ with respect to x_0 yields the following generalized eigenproblem

$$\mathcal{G}_o x_0 = \mathcal{G}_c^{-1}\epsilon(x_0)x_0.$$

And so $\epsilon(x_0)$ takes an extremal value for x_0 an eigenvector of $\mathcal{G}_c\mathcal{G}_o$ (or equivalently a generalized eigenvector of the pair $(\mathcal{G}_o, \mathcal{G}_c^{-1})$). This follows immediately from (2.1), while the maximization problem can be written as

$$\max_{x_0}\epsilon(x_0) = \max_{x_0}\frac{x_0^*\mathcal{G}_o x_0}{x_0^*\mathcal{G}_c^{-1}x_0},$$

or equivalently as

$$\max_{x_0}\epsilon(x_0) = \max_{x_0}\frac{x_0^*\mathcal{G}_c\mathcal{G}_o x_0}{x_0^* x_0},$$

which are respectively a generalized and a standard eigenproblem.
The extremal value of $\epsilon(.)$ corresponds thus to the maximal eigenvalue of $\mathcal{G}_c\mathcal{G}_o$ and hence to the square of the largest Hankel singular value σ_1 of the considered system.
Another interpretation is that the transformation T solves in fact the following minimization problem

$$\min_T \ Trace\ [T\mathcal{G}_c T^* + T^{-*}\mathcal{G}_o T^{-1}].$$

The minimum of this expression is $2\sum_{i=1}^{N}\sigma_i$, and balancing transformations turn out to provide a minimizing T [5].

The balancing transformation T ensures that each state is as controllable as it is observable in the new coordinate system. It is also shown in [91] that for non-minimal systems the controllable subspace and the unobservable subspace are the image and the kernel of $\mathcal{G}_c$ and $\mathcal{G}_o$, respectively. And so, T transforms the observability and controllability ellipsoids to an identical ellipsoid aligned with principle axes along the coordinate axes as shown in Figure 2.1.

[1] This means mainly that we have $rank\ (\mathcal{G}_c) = rank\ (\mathcal{G}_o) = N$.

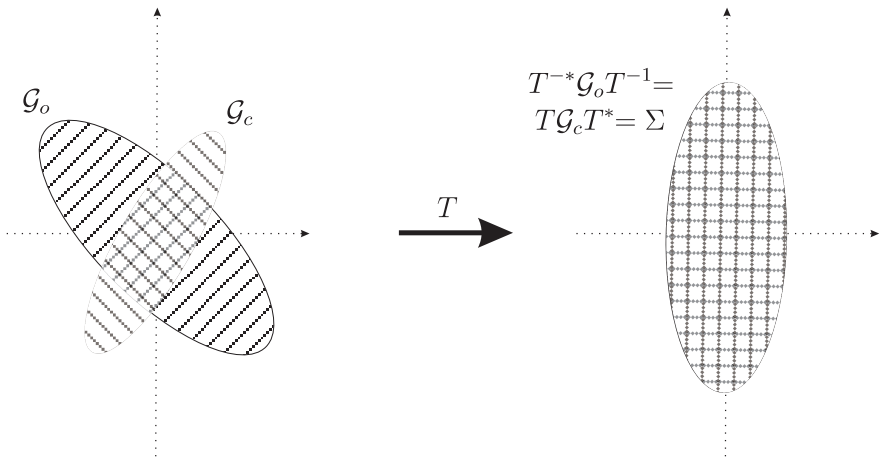


Fig. 2.1. The effect of a balancing transformation T on the controllability and observability ellipsoids.

After balancing the system, a reduced model is then obtained by truncating the new state $x = (x_1, \ldots, x_N)^T$ to $\hat{x} = (x_1, \ldots, x_n)^T$, $n < N$. This is equivalent to projecting the system with a rank n projection $\pi \doteq \pi_r \pi_l^*$. The so-called truncation matrices π_r and π_l can be obtained from the Cholesky factorizations of the matrices[2] (§.1.2.7)

$$G_c = S^* S \quad \text{and} \quad G_o = R^* R,$$

as follows. Compute the singular value decomposition :

$$SE^*R^* = \left[\, U_1 \middle| U_2 \,\right] \left[\begin{array}{c|c} \Sigma_1 & 0 \\ \hline 0 & \Sigma_2 \end{array}\right] \left[\, V_1 \middle| V_2 \,\right]^* \tag{2.2}$$

where $\Sigma_1 = diag\{\sigma_1, \ldots, \sigma_n\}$, $\Sigma_2 = diag\{\sigma_{n+1}, \ldots, \sigma_N\}$ and define

$$\pi_l = E^* R^* V_1 \Sigma_1^{-1/2}, \pi_r = S^* U_1 \Sigma_1^{-1/2}. \tag{2.3}$$

We can easily see that

$$\pi_l^* \pi_r = I_n \quad \text{(i.e., } \pi = \pi_r \pi_l^* \text{ is a projector) and} \quad \pi_l^* \mathcal{G}_c \mathcal{G}_o \pi_r = \Sigma_1^2.$$

It follows from this that the singular values σ_i of SE^*R^* are the (nonzero) Hankel singular values [137].
By this approach the Gramians $\mathcal{G}_c$ and $\mathcal{G}_o$ (and equivalently the matrices G_c and G_o) are not needed to construct the projector $\pi = \pi_r \pi_l^*$, but only the factors S and R, which can be obtained e.g. using Hammarling's method [68]. One then obtains the reduced model for the system $\mathcal{S} \doteq \{E, A, B, C\}$ as $\hat{\mathcal{S}} \doteq \{\pi_l^* E \pi_r, \pi_l^* A \pi_r, \pi_l^* B, C \pi_r\}$.

We summarize the procedure in the following algorithm.

[2] Note that G_c and G_o are related to the Gramians by $\mathcal{G}_c = G_c$, and $\mathcal{G}_o = E^* G_o E$. Furthermore they are solutions of (1.38) and (1.39).

Algorithm 3 The Balanced Truncation algorithm (BT).

• Solve for S and R using for example SLICOT functions

$$S^T = slstst(A, B, 0, 1), \quad \text{and} \quad R = slstst(A, C, 0, 0);$$

or

$$S^T = slgsst(A, E, B, 0, 1), \quad \text{and} \quad R = slgsst(A, E, C, 0, 0);$$

slstst : Solving stable Stein equations for the Cholesky factor of the solution,
slgsst : Solving stable generalized Stein equations for the Cholesky factor of the solution.

• Compute the SVD

$$SE^*R^* = U\Sigma V^*;$$

• The projection matrices are given by

$$\pi_l = E^*R^*V(:, 1:n)\left(\Sigma(1:n, 1:n)\right)^{-1/2},$$

and

$$\pi_r = S^*U(:, 1:n)\left(\Sigma(1:n, 1:n)\right)^{-1/2},$$

• And the reduced order model is given by the matrices

$$E_n = \pi_l^* E \pi_r, \quad A_n \pi_l^* A \pi_r, \quad B_n = \pi_l^* B, \quad \text{and} \quad C_n = C\pi_r.$$

An a priori error bounds in the induced 2-norm can be given for the error between the original and the reduced system [137]. The main result is that the $\mathcal{H}_\infty$-norm of the error system is bounded above by twice the sum of the neglected Hankel singular values :

$$\|\mathcal{S} - \hat{\mathcal{S}}\|_{\mathcal{H}_\infty} \leq 2(\sigma_{n+1} + \ldots + \sigma_N). \tag{2.4}$$

More recently, a new bound was derived by Antoulas [5] on the $\mathcal{H}_2$ norm. In the following subsection we give a discrete-time version of this bound.

2.2.1 $\mathcal{H}_2$ norm of the error system for Balanced Truncation

In this section we obtain a computable a posteriori upper bound for the $\mathcal{H}_2$ norm of the error system for Balanced Truncation[3].
For simplicity, let us assume that the matrix E is the identity[4] (i.e., $E = I_N$) and the system $\mathcal{S}$ is already in balanced form and that the matrices A, B and C are partitioned as follows :

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, B = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}, C = \begin{bmatrix} C_1 & C_2 \end{bmatrix},$$

where $\hat{A} \doteq A_{11} \in \mathbb{C}^{n\times n}$, $\hat{B} \doteq B_1 \in C^{n\times m}$ and $\hat{C} \doteq C_1 \in C^{p\times n}$. Since the system S is balanced its controllability and observability Gramians are diagonal and equal

$$\mathcal{G}_c = \mathcal{G}_o = \mathcal{G} = \begin{bmatrix} \mathcal{G}_1 & 0 \\ 0 & \mathcal{G}_2 \end{bmatrix}.$$

The unified Gramian $\mathcal{G}$ then solves the following Lyapunov equations

[3] This result was obtained with collaboration of Danny Sorensen who showed an equivalent bound for continuous case.
[4] Which means that we have a simple state-space model.

$$A\mathcal{G}A^* - \mathcal{G} + BB^* = 0, \tag{2.5}$$

$$A^*\mathcal{G}A - \mathcal{G} + C^*C = 0. \tag{2.6}$$

To obtain the result, we consider the error system

$$\mathcal{S}_e \doteq \left[\begin{array}{cc|c} A & 0 & B \\ 0 & A_{11} & -B_1 \\ \hline C & C_1 & 0 \end{array}\right],$$

with transfer function

$$G_e(z) = C(zI - A)^{-1}B - C_1(zI - A_{11})^{-1}B_1.$$

The bound on the approximation error $\|\mathcal{S} - \hat{\mathcal{S}}\|_{\mathcal{H}_2} = \|\mathcal{S}_e\|_{\mathcal{H}_2}$ is obtained directly by bounding the $\mathcal{H}_2$ norm of $\mathcal{S}_e$. Let us first note that the controllability Gramian $\mathcal{G}_{c_e}$ and the observability Gramian $\mathcal{G}_{o_e}$ of $\mathcal{S}_e$ are given by

$$\mathcal{G}_{c_e} = \begin{bmatrix} \mathcal{G} & -Y \\ -Y^* & \hat{\mathcal{G}}_c \end{bmatrix}, \quad \mathcal{G}_{o_e} = \begin{bmatrix} \mathcal{G} & Z \\ Z^* & \hat{\mathcal{G}}_o \end{bmatrix},$$

where $\hat{\mathcal{G}}_c$ and $\hat{\mathcal{G}}_o$ are the controllability and observability Gramian of the reduced model $\mathcal{S}_n$, respectively, and solve

$$A_{11}\hat{\mathcal{G}}_c A_{11}^* - \hat{\mathcal{G}}_c + B_1 B_1^* = 0, \tag{2.7}$$

$$A_{11}^*\hat{\mathcal{G}}_o A_{11} - \hat{\mathcal{G}}_o + C_1^* C_1 = 0, \tag{2.8}$$

and where $Z = \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix}$ and Y are solutions of

$$AYA_{11}^* - Y + BB_1^* = 0, \tag{2.9}$$

$$A^*ZA_{11} - Z + C^*C_1 = 0. \tag{2.10}$$

Hence, the $\mathcal{H}_2$ norm of the error system is given by 1.2.8

$$\begin{aligned} \|\mathcal{S}_e\|_{\mathcal{H}_2}^2 &= Trace\left\{ \begin{bmatrix} B^* & -B_1^* \end{bmatrix} \begin{bmatrix} \mathcal{G} & Z \\ Z^* & \hat{\mathcal{G}}_o \end{bmatrix} \begin{bmatrix} B \\ -B_1 \end{bmatrix} \right\} \\ &= Trace\left\{ B^*\mathcal{G}B - 2B^*ZB_1 + B_1^*\hat{\mathcal{G}}_o B_1 \right\} \\ &= Trace\left\{ B^*\mathcal{G}B - 2B_1^*Z_1B_1 - 2\underline{B_2^*Z_2B_1} + B_1^*\hat{\mathcal{G}}_o B_1 \right\}. \end{aligned}$$

Now, from (2.5), we obtain

$$A_{11}\mathcal{G}_1A_{21}^* + A_{12}\mathcal{G}_2A_{22}^* + B_1B_2^* = 0,$$

and consequently

$$\begin{aligned} Trace\left\{-2B_2^*Z_2B_1\right\} &= Trace\left\{-2B_1B_2^*Z_2\right\} \\ &= Trace\left\{2A_{11}\mathcal{G}_1A_{21}^*Z_2 + 2A_{12}\mathcal{G}_2A_{22}^*Z_2\right\}. \end{aligned}$$

Substituting yields

$$\|\mathcal{S}_e\|_{\mathcal{H}_2}^2 = Trace\left\{B^*\mathcal{G}B - 2B_1^*Z_1B_1 + \underline{2A_{11}\mathcal{G}_1A_{21}^*Z_2} + 2A_{12}\mathcal{G}_2A_{22}^*Z_2 + B_1^*\hat{\mathcal{G}}_oB_1\right\}.$$

From (2.10), we have

$$A_{11}^*Z_1A_{11} + A_{21}^*Z_2A_{11} - Z_1 + C_1^*C_1 = 0,$$

and consequently

$$\begin{aligned} Trace\left\{2A_{11}\mathcal{G}_1A_{21}^*Z_2\right\} &= Trace\left\{2\mathcal{G}_1A_{21}^*Z_2A_{11}\right\} \\ &= Trace\left\{-2\mathcal{G}_1A_{11}^*Z_1A_{11} + 2\mathcal{G}_1Z_1 - 2\mathcal{G}_1C_1^*C_1\right\}. \end{aligned}$$

Combining this with the definition of the $\mathcal{H}_2$ norm of $\mathcal{S}$ and $\hat{\mathcal{S}}$

$$\|\mathcal{S}\|_{\mathcal{H}_2}^2 = Trace\left\{B^*\mathcal{G}B\right\} = Trace\left\{C\mathcal{G}C^*\right\},$$

and

$$\|\hat{\mathcal{S}}\|_{\mathcal{H}_2}^2 = Trace\left\{B_1^*\hat{\mathcal{G}}_oB_1\right\} = Trace\left\{C_1\hat{\mathcal{G}}_cC_1^*\right\},$$

gives

$$\begin{aligned} \|\mathcal{S}_e\|_{\mathcal{H}_2}^2 = Trace\Big\{&2A_{12}\mathcal{G}_2A_{22}^*Z_2 + C_2\mathcal{G}_2C_2^* - C_1\mathcal{G}_1C_1^* + C_1\hat{\mathcal{G}}_cC_1^*\Big\} \\ &+Trace\left\{\underline{-2B_1B_1^*Z_1 - 2A_{11}\mathcal{G}_1A_{11}^*Z_1 + 2\mathcal{G}_1Z_1}\right\}. \end{aligned}$$

Now, back to (2.5) and taking the first block leads to the equation

$$A_{11}\mathcal{G}_1A_{11}^* + A_{12}\mathcal{G}_2A_{12}^* - \mathcal{G}_1 + B_1B_1^* = 0,$$

from which it follows

$$Trace\left\{-2B_1B_1^*Z_1 - 2A_{11}\mathcal{G}_1A_{11}^*Z_1 + 2\mathcal{G}_1Z_1\right\} = Trace\left\{2A_{12}\mathcal{G}_2A_{12}^*Z_1\right\}.$$

And finally, we obtain

$$\|\mathcal{S}_e\|_{\mathcal{H}_2}^2 = Trace\left\{C_2\mathcal{G}_2C_2^* + C_1(\hat{\mathcal{G}}_c - \mathcal{G}_1)C_1^* + 2A_{12}\mathcal{G}_2\begin{bmatrix} A_{12}^* & A_{22}^* \end{bmatrix}\begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix}\right\}.$$

From the Cauchy-Schwartz inequality we obtain

$$\left|Trace\left\{C_2\mathcal{G}_2C_2^*\right\}\right| \leq \sigma_{n+1}.\|C_2\|_2^2,$$

$$\left|Trace\left\{C_1(\hat{\mathcal{G}}_c - \mathcal{G}_1)C_1^*\right\}\right| \leq \|\hat{\mathcal{G}}_c - \mathcal{G}_1\|_2.\|C_1\|_2^2,$$

$$\left|Trace\left\{2A_{12}\mathcal{G}_2\begin{bmatrix} A_{12}^* & A_{22}^* \end{bmatrix}\begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix}\right\}\right| \leq 2\sigma_{n+1}.\|A_{12}\|_2\left\|\begin{bmatrix} A_{21} \\ A_{22} \end{bmatrix}\right\|_2\|Z\|_2.$$

As Z is the solution of the Sylvester equation (2.9), it has the form

$$Z = \sum_{i=0}^{\infty}(A^*)^iC^*C_1A_{11}^i,$$

and so

$$\|Z\|_2 \leq \|C\|_2^2 \sum_{i=0}^{\infty} \|A^i\|_2 . \|A_{11}^i\|_2 .$$

We summarize this discussion in the following result.

Theorem 2.3.

$$\|\mathcal{S} - \hat{\mathcal{S}}\|_{\mathcal{H}_2}^2 \leq c.\sigma_{n+1}$$

where

$$c = \|C\|_2^2 . \left(1 + \frac{\|\hat{\mathcal{G}}_c - \mathcal{G}_1\|_2}{\sigma_{n+1}} + 2\|A_{12}\|_2 \| \begin{bmatrix} A_{21} \\ A_{22} \end{bmatrix} \|_2 \sum_{i=0}^{\infty} \|A^i\|_2 . \|A_{11}^i\|_2 \right).$$

■

Remark 2.4.

- $E = \hat{\mathcal{G}}_c - \mathcal{G}_1$ satisfies the Sylvester equation

$$A_{11}^* E A_{11} - E + A_{21}^* \mathcal{G}_2 A_{21} = 0.$$

- For balanced systems, we have :

$$\|C\|_2 = \|B\|_2 \quad \text{and} \quad \|\hat{\mathcal{G}}_o - \mathcal{G}_1\|_2 = \|\hat{\mathcal{G}}_c - \mathcal{G}_1\|_2,$$

  and so c is also given by

$$c = \|B\|_2^2 . \left(1 + \frac{\|\hat{\mathcal{G}}_o - \mathcal{G}_1\|_2}{\sigma_{n+1}} + 2\|A_{12}\|_2 \| \begin{bmatrix} A_{21} \\ A_{22} \end{bmatrix} \|_2 \sum_{i=0}^{\infty} \|A^i\|_2 . \|A_{11}^i\|_2 \right).$$

2.2.2 Time-varying Balanced Truncation

For linear time-varying systems, the procedure is analogous to the linear time-invariant case but requires the solution of two sets of difference equations in order to obtain the Gramians at each time step. Moreover the system must be uniformly controllable and observable over the considered interval [107, 133]. For discrete-time systems this means that there exists a time-varying state-space transformation T_k (uniformly bounded with uniformly bounded inverse) so that the transformed Gramians satisfy

$$T_k^{-1} \mathcal{G}_c(k) \mathcal{G}_o(k) T_k = \tilde{\mathcal{G}}_c(k) \tilde{\mathcal{G}}_o(k) = \Sigma^2(k),$$
$$0 < \Sigma(k) < \infty I.$$

One also shows that if there is a uniform gap between "large" and "small" singular values, the constructed reduced model is asymptotically stable, and uniformly controllable and observable [107]. Rather than computing the complete transformation T_k, one can try to estimate only the first few columns of T_k, i.e., a matrix $X_k \in \mathbb{R}^{N \times n}$ whose columns span the "dominant" eigenvectors of the product $\mathcal{G}_c(k)\mathcal{G}_o(k)$. Similarly, one will need a matrix $Y_k \in \mathbb{R}^{N \times n}$ whose columns span the "dominant" eigenvectors of the product $\mathcal{G}_o(k)\mathcal{G}_c(k)$.
One then obtains the reduced model for the system $\{E_k, A_k, B_k, C_k\}$ as

$$\{Y_k^* E_k X_k, Y_k^* A_k X_k, Y_k^* B_k, C_k X_k\}.$$

If $E_k = I$ for all k, the bases provided have to be normalized using $Y_k^T X_k = I_n$ (see [128] for more details).

Recently, a similar a priori error bound to (2.4) for time-varying systems was obtained in [106].

Theorem 2.5. *[106] Given a balanced time-varying system $\mathcal{S}$ on the interval $[0,T]$ where each diagonal element $\sigma_i(k)$, $i \in [n+1, N]$ of $\Sigma(k)$, is either non-increasing or non-decreasing for all $k \in [0,T]$. We have*

$$\|\mathcal{S} - \mathcal{S}_n\| \leq 2 \sum_{i=n+1}^{N} \sup_{k\in[0,T]} \sigma_i(k),$$

where $\mathcal{S}_n$ is the truncated system of order n.

Either time-invariant or time-varying Balanced Truncation has the property that n HSV are retained. Next, we investigate methods that preserve other parameters of a system.

2.3 Krylov-based approximation methods

In this section we discuss matching of the transfer function at specific frequencies using a projection. This can be done by matching the Markov parameters [46] or the moments at particular points of the complex plane. This matching at a finite number of frequencies is a form of interpolation [57]. Given a system $\mathcal{S}$, we can expand its transfer function around z_0 :

$$T_f(z) = \eta_0 + \eta_1 \frac{(z-z_0)}{1!} + \eta_2 \frac{(z-z_0)^2}{2!} + \dots$$

The approximation problem consists in finding $\hat{S}$, which has a transfer function

$$\hat{T}_f(z) = \hat{\eta}_0 + \hat{\eta}_1 \frac{(z-z_0)}{1!} + \hat{\eta}_2 \frac{(z-z_0)^2}{2!} + \dots$$

such that for appropriate k :

$$\eta_i = \hat{\eta}_i, \quad i = 1, 2, \dots, k.$$

Moment matching methods can be implemented in a numerically stable and efficient way. Matching Markov parameters is known as *partial realization*. When $z_0 = 0$, the corresponding problem is known as *Padé approximation*. If z_0 takes a finite number of points ω_i, it is called a *multi-point Padé approximation*. In the general case, the problem is known as *rational interpolation*. Rational interpolation generally results in a good approximation of the original transfer function near the expansion points, but may not be accurate at other frequencies.
In many cases the computation of the moments is numerically problematic, so we have to use another family of algorithms known as rational Krylov methods, in particular, Lanczos and Arnoldi methods. Two important properties of these methods are :

- moment matching is achieved without explicit computation of the moments, and
- the computation is implemented iteratively.

In the previous section, the construction of the projector was based on a SVD of the product of the Cholesky factors of the Gramians (2.2), which are dense matrices even if the system matrices are sparse.
Krylov methods are used to approximate the dominant eigenspace of the product of the Gramians $\mathcal{G}_c\mathcal{G}_o$. They are used in fact to construct an n dimensional basis with $n \ll N$ and yet capture the dominant features of both Gramians.
The link is made using the Hankel map :

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \end{bmatrix} = \underbrace{\begin{bmatrix} CE^{-1} \\ CE^{-1}(AE^{-1}) \\ CE^{-1}(AE^{-1})^2 \\ \vdots \end{bmatrix}}_{\mathcal{O}_E} E \underbrace{\begin{bmatrix} E^{-1}B & (E^{-1}A)E^{-1}B & (E^{-1}A)^2E^{-1}B & \dots \end{bmatrix}}_{\mathcal{C}_E} \begin{bmatrix} u_{-1} \\ u_{-2} \\ u_{-3} \\ \vdots \end{bmatrix}.$$

From this, one constructs Krylov sequences to approximate both factors $\mathcal{O}_E$ and $\mathcal{C}_E$.

Suppose now that we want to match the transfer function $G(z) = C(zE - A)^{-1}B$ at $k^l + k^r$ distinct frequencies $\omega_{l,1}, \dots, \omega_{l,k^l}, \omega_{r,1}, \dots, \omega_{r,k^r}$, where the subscripts l and r stand for "left" and "right". To achieve the matching, we can use the projection matrix $\pi = \pi_r \pi_l^*$ ($\pi_l^* \pi_r = I_n$) of rank n such that :

$$\pi(e^{j\omega_{l,i}}E - A)^{-1}B = (e^{j\omega_{l,i}}E - A)^{-1}B, \quad \text{for} \quad i = 1, \dots, k^l \tag{2.11}$$

$$C(e^{j\omega_{r,i}}E - A)^{-1}\pi = C(e^{j\omega_{r,i}}E - A)^{-1}, \quad \text{for} \quad i = 1, \dots, k^r \tag{2.12}$$

then with $E_n = \pi_l^* E \pi_r$, $A_n = \pi_l^* A \pi_r$, $B_n = \pi_l^* B$, $C_n = C\pi_r$, we have :

$$C_n(e^{j\omega_i}E_n - A_n)^{-1}B_n = C(e^{j\omega_i}E - A)^{-1}B, \quad \forall \omega_i \in \{\omega_{l,i}\} \cup \{\omega_{r,i}\}.$$

Here the rank n of the projection is determined from the number of frequency points for which matching is required.

The projection matrix π can be constructed as follows :
First the equations (2.11) and (2.12) can be rewritten as

$$\pi \mathcal{C}_\omega = \mathcal{C}_\omega, \quad \mathcal{O}_\omega \pi = \mathcal{O}_\omega \tag{2.13}$$

where

$$\mathcal{C}_\omega = \left[(e^{j\omega_{l,1}}E - A)^{-1}B \dots (e^{j\omega_{l,k^l}}E - A)^{-1}B \right],$$

and

$$\mathcal{O}_\omega = \begin{bmatrix} C(e^{j\omega_{r,1}}E - A)^{-1} \\ \vdots \\ C(e^{j\omega_{r,k^r}}E - A)^{-1} \end{bmatrix}. \tag{2.14}$$

Let $\mathcal{C}_\omega = \pi_1 X$ and $\mathcal{O}_\omega^* = \pi_2 Y$ where X and Y are nonsingular and so that

$$\pi_2^* \pi_1 = I.$$

From (2.13) it follows that

$$\pi\pi_1 = \pi_1, \quad \pi_2^*\pi = \pi_2^*,$$

and so $\pi = \pi_1\pi_2^*$ is the projector.
We have the following theorems which connect projection via Krylov subspaces and either the matching of Markov parameters (Theorem. 2.6) or the matching of moments at the points $\omega_1, \dots, \omega_k (\neq \infty)$ (Theorem. 2.7) [64].

Theorem 2.6. *Let us define*

$$\mathcal{K}_{k^r}(E^{-1}A, E^{-1}B) = span\{E^{-1}B, E^{-1}AE^{-1}B, \dots, (E^{-1}A)^{k^r-1}E^{-1}B\},$$

and

$$\mathcal{K}_{k^l}(E^{-*}A^*, E^{-*}C^*) = span\{E^{-*}C^*, E^{-*}C^*E^{-*}A^*, \ldots, E^{-*}C^*(E^{-*}A^*)^{k^l-1}\}.$$

Now, if

$$\mathcal{K}_{k^r}(E^{-1}A, E^{-1}B) \subseteq Im(\pi_r),$$

and

$$\mathcal{K}_{k^l}(E^{-*}A^*, E^{-*}C^*) \subseteq Im(\pi_l),$$

then

$$C(E^{-1}A)^{i-1}E^{-1}B = C_n(E_n^{-1}A_n)^{i-1}E_n^{-1}B_n, \quad for \quad i = 1, 2, \ldots, k^l + k^r.$$

■

Theorem 2.7. *If*

$$\bigcup_{i=1}^{k} \mathcal{K}_{k_i^r}((A - e^{j\omega_i}E)^{-1}E, (A - e^{j\omega_i}E)^{-1}B) \subseteq Im(\pi_r),$$

and

$$\bigcup_{i=1}^{k} \mathcal{K}_{k_i^l}((A - e^{j\omega_i}E)^{-T}E^*, (A - e^{j\omega_i}E)^{-T}C^*) \subseteq Im(\pi_l),$$

then,

$$C\left((A - e^{j\omega_i}E)^{-1}E\right)^{j_i-1}(A - e^{j\omega_i}E)^{-1}B = C_n\left((A_n - e^{j\omega_i}E_n)^{-1}E_n\right)^{j_i-1}(A_n - e^{j\omega_i}E_n)^{-1}B_n,$$

for $j_i = 1, 2, \ldots, k_i^r + k_i^l$ *and* $i = 1, 2, \ldots, k$. ■

For the moment matching methods a very important open question that remains is how to choose the moment matching points ω_i, and their order k_i, so that the global approximation error is small. Many authors proposed some heuristic choices [64], such that by trial and error, the interpolation points are chosen to minimize the errors in the interpolation points.

In general, strict stability of the reduced model cannot be guaranteed even if the original model is stable. Moreover, these methods may turn a strictly stable system into an unstable system. This problem can be fixed by *implicit restart methods* [67].

Stability can be guaranteed by decoupling the matching problem [136]. The idea is to use either $\mathcal{C}_\omega\mathcal{C}_\omega^*$, $\mathcal{G}_o$ or $\mathcal{G}_c$, $\mathcal{O}_\omega^*\mathcal{O}_\omega$ in defining the projection matrix π. For example consider the observability Gramian $\mathcal{G}_o$, and let $n = k^r p$ be the rank of the matrix $\mathcal{O}_\omega$ (2.14), then there exists a unitary state-space transformation U such that

$$\bar{\mathcal{G}}_o = U^*\mathcal{G}_o U = U^*\mathcal{O}_\omega^*\mathcal{O}_\omega U = \begin{bmatrix} \bar{\Sigma}_n & 0 \\ 0 & 0 \end{bmatrix}.$$

Now if we take the realization of our system $\mathcal{S} = \{A, B, C\}$ for which

$$\mathcal{G}_o = \begin{bmatrix} \Sigma_n & 0 \\ 0 & 0 \end{bmatrix}, \quad \mathcal{O}_\omega = \begin{bmatrix} \Sigma_n^{1/2} & 0 \end{bmatrix}, \quad \text{and} \quad \mathcal{G}_c = \begin{bmatrix} \mathcal{G}_{c_{11}} & \mathcal{G}_{c_{12}} \\ \mathcal{G}_{c_{12}}^* & \mathcal{G}_{c_{22}} \end{bmatrix},$$

where $\Sigma_n = U\bar{\Sigma}_n U^*$. Then

$$\mathcal{S}_n = \left[\begin{array}{c|c} T^{-1}AT & T^{-1}B \\ \hline CT & \end{array}\right] \quad \text{with} \quad T = \left[\begin{array}{c|c} I_n & 0 \\ \hline \mathcal{G}_{c_{12}}^*\mathcal{G}_{c_{11}}^{-1} & I_{N-n} \end{array}\right]$$

has a block-diagonal controllability Gramian and an observability Gramian equal to $\mathcal{G}_o$. The n truncation yields a strictly stable system since $T^{-1}\mathcal{G}_c T^{-*}$ is block diagonal. The projector is

$$\pi = \begin{bmatrix} I_n & 0 \\ \mathcal{G}_{c12}^{*}\mathcal{G}_{c11}^{-1} & 0 \end{bmatrix}$$

and satisfies $\mathcal{O}\pi = \mathcal{O}$.

Finally, it should be stressed that the Krylov based methods have reliable numerical implementations, which involve matrix-vector multiplications, exclusively [64]. They can be applied to large scale systems, unlike Balanced Truncation.

Recently, there has been a new approach based on a combination of Balanced Truncation and Krylov based methods, that could be called *recursive low-rank Balanced Truncation* or *Approximated Balanced Truncation* (ABT). The idea is not new : there exist so-called Smith methods (see e.g. the work of Gugercin-Sorenson-Antoulas [67], R.Li [84]). In the next section we give a description of the Approximated Balanced Truncation method.

2.4 Approximated Balanced Truncation

First, notice that the Balanced Truncation procedure (Algorithm 3) is based on the square roots of the Gramians. These square roots can be well approximated by low-rank approximations. The idea of Approximated Balanced Truncation is to use these low-rank approximations instead of the original square roots to provide an approximation to Balanced Truncation (see §.2.2).
Notice that even if the low-rank approximations were obtained from a discretization of the system, i.e., the discretized Gramians, any low-rank approximation of the discretized Gramian would be also a low-rank approximation of the continuous-time Gramian since they are preserved under the bilinear transformation. Actually since the two Gramians are equal, their dominant singular subspace are also equal. This property is used to obtain a reduced model of a continuous-time system whose projection matrices are computed from the discretized version of this system. The implemented algorithm is given below.

Algorithm 4 The Approximated Balanced Truncation algorithm (ABT).

Required : The low-rank approximations $S_i, R_i \in \mathbb{R}^{N\times n}$ of the Gramians $\mathcal{G}_c$ and $\mathcal{G}_o$;

- Calculate the singular value decomposition of $S_i^T R_i$

$$S_i^T R_i = U\Sigma V^T;$$

- Let

$$X = S_i U \Sigma^{-1/2}, \quad \text{and} \quad Y = R_i V \Sigma^{-1/2};$$

- The order n approximated truncated balanced realization is given by :

$$A_n = Y^* A X, \quad B_n = Y^* B, \quad \text{and} \quad C_n = CX.$$

Remark 2.8.

Here we use the SVD to "balance" the projection matrices. This is crucial because we approximate independently the Gramians. In practice, if the system has poles close to the unit circle, one or both Gramians are not well approximated. So we use the SVD to balance the error. We obtain indeed a convenient reduced-order model. This alternative balancing method was also proposed by Varga in [130]. He called it *balancing-free square-root method*, and its advantage is that it has a potentially better numerical accuracy for systems that are poorly scaled originally.

Now, let us discuss the effect of the approximation of Gramians on the quality of the obtained reduced model [67].

We consider the n^{th} order reduced system obtained by Balanced Truncation,

$$\mathcal{S}_n^{BT} = \left(\begin{array}{c|c} A_n^{BT} & B_n^{BT} \\ \hline C_n^{BT} & \end{array}\right) = \left(\begin{array}{c|c} \pi_l^* A \pi_r & \pi_l^* B \\ \hline C\pi_r & \end{array}\right),$$

where π_l and π_r are constructed as in (§.2.2). Similarly, let

$$\mathcal{S}_n = \left(\begin{array}{c|c} A_n & B_n \\ \hline C_n & \end{array}\right) = \left(\begin{array}{c|c} Y^* A X & Y^* B \\ \hline CX & \end{array}\right)$$

be the n^{th} order reduced model obtained by an Approximated Balanced Truncation where approximate Cholesky factors are used instead of the exact Cholesky factors for computing the projection matrices X and Y. The following equation is then easily derived :

$$A_n \Sigma A_n^* + B_n B_n^* - \Sigma = Y^* \Delta Y - Y^* A \Delta A^* Y$$

where Δ is the error in the Gramian $\mathcal{G}_c$, i.e.,

$$\Delta \doteq \mathcal{G}_c - S_i S_i^T,$$

and Σ is the diagonal matrix obtained in Algorithm 4. The diagonal elements of the matrix Σ are in fact a perturbation of the n HSV of the system $\mathcal{S}_n = \{A_n, B_n, C_n\}$ and also of the n dominant HSV of the system $\mathcal{S} = \{A, B, C\}$. This perturbation depends mainly of Δ. It is clear that the stability of the reduced system is not always guaranteed[5]. However, this does not seem to occur often in practice (see also [67]); in general we obtain a stable reduced system for each of our computational examples.
But notice that one can use the idea presented in Section 2.3 to stabilize the resulting reduced order model if it is unstable.

The following result examines how close S_n^{BT} is to $\mathcal{S}_n$.

Corollary 2.9. *Assume that X and Y are close to π_r and π_l respectively, i.e., there exists a small number ϵ s.t.*

$$\|\pi_r - X\| \leq \epsilon, \quad \textit{and} \quad \|\pi_l - Y\| \leq \epsilon.$$

Then we have :

$$\begin{array}{l} \|\mathcal{S}_n^{BT} - \mathcal{S}_n\|_\infty \leq \\ \epsilon \left(\|C_n^{BT}\| \|B_n^{BT}\| \|A_n^{BT}\| \left(\|\pi_l\| + \|\pi_r\|\right) + \|\mathcal{S}_1\|_\infty \|B_n^{BT}\| + \|\mathcal{S}_2\|_\infty \|C_n^{BT}\| \right) + O(\epsilon^2) \end{array}$$

where

$$\mathcal{S}_1 \doteq \left(\begin{array}{c|c} A_n^{BT} & I \\ \hline C_n^{BT} & \end{array}\right), \quad \textit{and} \quad \mathcal{S}_2 \doteq \left(\begin{array}{c|c} A_n^{BT} & B_n^{BT} \\ \hline I & \end{array}\right).$$

■

Proof. In order to prove this result, let us define first :

$$\Delta_r \doteq \pi_r - X, \quad \text{and} \quad \Delta_l \doteq \pi_l - Y,$$

and let

[5] We have the same kind of equation for the observability.

$$\|\Delta_r\| \leq \epsilon, \quad \text{and} \quad \|\Delta_l\| \leq \epsilon$$

where ϵ is a small number. If we consider the following matrices

$$\Delta_A \doteq A_n^{BT} - A_n, \quad \Delta_B \doteq B_n^{BT} - B_n \quad \text{and} \quad \Delta_C \doteq C_n^{BT} - C_n.$$

We have

$$\Delta_A = \pi_l^* A \pi_r - Y^* A X = \pi_l^* A(\pi_r - X) + (\pi_l - Y)^* A X = \pi_l^* A \Delta_r - \Delta_l^* A X,$$

$$\Delta_B = \pi_l^* B - Y^* B = \Delta_l^* B, \quad \text{and} \quad \Delta_C = C\pi_r - CX = C\Delta_r.$$

And so Δ_A, Δ_B and Δ_C satisfy

$$\|\Delta_A\| \leq \epsilon \|A\| \left(\|\pi_l\| + \|\pi_r\|\right) + \epsilon^2 \|A\|,$$

$$\|\Delta_B\| \leq \epsilon \|B\|, \quad \text{and} \quad \|\Delta_C\| \leq \epsilon \|C\|.$$

To simplify the discussion, we shall assume that

$$(j\omega - A_n)^{-1} \approx (j\omega - A_n^{BT})^{-1} + \hat{\Delta}_A$$

holds for every $\omega \in \mathbb{R}$ where $\hat{\Delta}_A = (j\omega - A_n^{BT})^{-1} \Delta_A (j\omega - A_n)^{-1}$ satisfies the same upper bound as Δ_A, i.e.,

$$\|\hat{\Delta}_A\| \leq \epsilon \|A\| \left(\|\pi_l\| + \|\pi_r\|\right) + \epsilon^2 \|A\|. \tag{2.15}$$

Now, if we consider the $\mathcal{H}_\infty$ norm of the error system $\mathcal{S}_n^{BT} - \mathcal{S}_n$ we have

$$\begin{aligned} \mathcal{S}_n^{BT} - \mathcal{S}_n &= T_f^{BT} - T_{f_n} \\ &= C_n^{BT} \left(e^{j\omega} I - A_n^{BT}\right)^{-1} B_n^{BT} - C_n \left(e^{j\omega} I - A_n\right)^{-1} B_n \end{aligned}$$

Using (2.15) and the definitions of Δ_A, Δ_B and Δ_C we obtain

$$\begin{aligned} \|\mathcal{S}_n^{BT} - \mathcal{S}_n\|_{\mathcal{H}_\infty} &= \|C_n^{BT} T_A B_n^{BT} - (C_n^{BT} - \Delta_C) \left[T_A + \hat{\Delta}_A\right] (B_n^{BT} - \Delta_B)\|_2 \\ &= \|-\Delta_C T_A B_n^{BT} - C_n^{BT} T_A \Delta_B + (C_n^{BT} - \Delta_C)\hat{\Delta}_A (B_n^{BT} - \Delta_B)\|_2 \end{aligned}$$

where $T_A = \left(e^{j\omega} I - A_n^{BT}\right)^{-1}$. Finally, using

$$\mathcal{S}_1 \doteq \left(\begin{array}{c|c} A_n^{BT} & I \\ \hline C_n^{BT} & \end{array}\right), \quad \text{and} \quad \mathcal{S}_2 \doteq \left(\begin{array}{c|c} A_n^{BT} & B_n^{BT} \\ \hline I & \end{array}\right),$$

it is easy to deduce the final result

$$\begin{aligned} &\|\mathcal{S}_n^{BT} - \mathcal{S}_n\|_\infty \leq \\ &\quad \epsilon \left(\|C\|\|B\|\|A\| \left(\|\pi_l\| + \|\pi_r\|\right) + \|\mathcal{S}_1\|_\infty \|B\| + \|\mathcal{S}_2\|_\infty \|C\|\right) + O(\epsilon^2). \end{aligned}$$

□

Hence for small ϵ, i.e., when X and Y are, respectively, close to π_r and π_l, we expect $\mathcal{S}_n^{BT}$ to be close $\mathcal{S}_n$. This result says that the quality of a reduced order model depends on the distance between the projection matrices and those of BT. In [67], this result was given formally without proof for the continuous-time case. Here we gave a proof for the discrete-time case, but this may not say much about the quality of approximations. In general, the choice of coordinate system for A_n, B_n and C_n plays an important role as well.

2.5 Concluding remarks

In this chapter we review the most used projection based methods in model reduction, especially Balanced Truncation and Krylov subspaces methods. We also present a new derivation of an a posteriori bound of the $\mathcal{H}_2$ norm of the error system corresponding to the Balanced Truncation method.
We introduce a new method based on a combination of Krylov subspace ideas and the Balanced Truncation procedure. We called it *Approximated Balanced Truncation*.

The idea of this method is based on the fact that most techniques for model reduction of linear systems are based on the dominant subspaces of Gramians. In applications typically there is a rapid decay in the Gramians eigenvalues, and so these Gramians can be well approximated using low-rank approximations. These low-rank approximations are used instead of the original Gramians in the Balanced Truncation procedure to provide the reduced order model. This method has very nice properties. The combination of Krylov subspace ideas and the Balanced Truncation procedure implies that Approximated Balanced Truncation inherits the nice properties of both methods.
It benefits of the iterative computations inherited from Krylov subspace ideas. This iterative computations will reduce significatively the cost and make use of any sparsity in the data. The use of the Balanced Truncation procedure yields bounds on the quality of the approximations.

In the next four chapters, we present three novel algorithms which can be used to produce recursively the low-rank approximations of the Gramians. The complexities of these algorithms are of the order of $O(N)$, where N is the order of the original system.

Chapter 3

Low-rank approximation methods

Most techniques for model reduction of linear systems are based on the dominant subspaces of Gramians (energy functions for in- and outgoing signals) or the dominant subspaces of their product. These Gramians are the solutions of Stein equations. Computing efficiently these solutions (or their dominant subspaces) when the system matrices are large and sparse is still very difficult (see for instance [17, 18, 19] for a survey). In fact, direct methods ignore sparsity in the Stein equations and are not very attractive for parallelization. For example, even if BT has the nice property that from a stable model it produces a reduced model which is guaranteed to be stable with a global a priori $\mathcal{H}_\infty$-error bound, in general the Krylov based methods are to be preferred for large problems.
The high complexity of BT is due to the fact that we solve two Stein equations and then compute a SVD, which both have complexity $O(N^3)$. And so for systems with $N \gg 100$ the cost of BT becomes prohibitive. The complexity of the "square root" method is still prohibitive due to the full balancing SVD (2.2).
However, if the Cholesky factors have low-rank the computational cost will be significantly reduced.

Recently, Penzl and others [98, 7] have observed that solutions to Stein (and equivalently Lyapunov) equations associated with LTI systems often have low numerical rank, which means that there is a rapid decay in the Gramians eigenvalues, and so a rapid decay in the HSV.
Indeed, the idea of low-rank methods is to take advantage of this low-rank structure to obtain approximate solutions in low-rank factored form. The principal outcome of these approaches is that the complexity and the storage are reduced from $O(N^3)$ flops and $O(N^2)$ words of memory to $O(N^2 r)$ flops and $O(Nr)$ words of memory, respectively, where r is the "approximate" rank of the Gramian ($r \ll N$). In fact, these low-rank schemes are the only way to solve efficiently very large scale Stein equations. Moreover, approximating directly the Cholesky factors of the Gramians and using these approximations to provide a reduced model, has a comparable cost to that of the popular moment matching methods. It requires only matrix-vector products and linear solvers.

In the sequel, we show how to approximate recursively the Gramians by a low-rank factorization or equivalently to approximate their Cholesky factors by a low-rank approximation, and at the same time exploit the possible sparsity of the model matrices.
Although all material below should be applied to both Gramians $\mathcal{G}_c$ and $\mathcal{G}_o$, we will focus on the controllability Gramian $\mathcal{G}_c$ only. Of course, the ideas developed here apply equally well to computing the observability Gramian $\mathcal{G}_o$.

3.1 Smith method

A popular approach for the time-invariant case is the *Smith method*. It was analyzed by Penzl [95]. This method computes recursive approximations to the solution of the Stein equation :

$$\mathcal{G}_c(k+1) = A\mathcal{G}_c(k)A^* + BB^*,$$

which yields, for $\mathcal{G}_c(0) = 0$, the iterates :

$$\mathcal{G}_c(k) = \sum_{i=0}^{k-1} \Phi(k, i+1)BB^*\Phi(k, i+1)^*,$$

converging to the solution $\mathcal{G}_c$ of

$$\mathcal{G}_c = A\mathcal{G}_c A^* + BB^*, \tag{3.1}$$

where $\Phi(.,.)$ is the transition matrix which can be defined over any time period $[i, j]$ by

$$\Phi(j, i) \doteq A_{j-1}A_{j-2}\dots A_{i+1}A_i.$$

The Smith method converges linearly but has the drawback that it computes the solution in dense form and hence requires $O(N^2)$ storage and $O(N^3)$ per iteration if A is dense [67].
If the discrete-time system considered is obtained from a continuous system using the bilinear transformation (1.2.9), accelerated versions of this method have been proposed by Penzl in [97], called the *Multi-shift Smith*, *squared Smith* and *cyclic Smith(l)* method. The key idea is to apply the Smith method cyclically with a finite number of shift parameters $\{\mu_1, \dots, \mu_l\}$ (for more details see [97]). But, this method still has the storage requirement $O(N^2)$.
To take advantage of the "approximate" low-rank property of the Gramian, one has to use the so-called *low-rank Smith method*.

3.2 Low-rank Smith method

The key idea of the low-rank method is to substitute the iterates by their symmetric factorization :

$$\mathcal{G}_c(k) = S_k S_k^*.$$

This is always possible because $\mathcal{G}_c(k)$ can be shown to be symmetric and positive semi-definite (§.1.2.5). And instead of explicitly forming $\mathcal{G}_c$ or its square root S_k, the low-rank methods compute and store the low-rank approximate square root factor, which corresponds in this case to the controllability matrix $\mathcal{C}_k$ [97]. And so the original Gramian $\mathcal{G}_c(k)$ is approximated by $\mathcal{P}_k^S = \mathcal{C}_k\mathcal{C}_k^*$, where the superscript S refers to Smith. The low-rank Smith method consists of two steps. First, let us define

$$\mathcal{C}^{(1)} = B,$$

we initialize S_1^S by :

$$\mathcal{C}_1 = B,$$

followed by the actual low-rank Smith iteration :

$$\begin{aligned} \mathcal{C}^{(i+1)} &= A\mathcal{C}^{(i)}, \\ \mathcal{C}_{i+1} &= \left[\mathcal{C}_i \ \mathcal{C}^{(i+1)}\right]. \end{aligned} \tag{3.2}$$

It then follows that :

$$\mathcal{C}_k = \left[B \ AB \ A^2B \ \dots \ A^{k-1}B\right]. \tag{3.3}$$

We note that at the k^{th} step, $\mathcal{C}_k$ has $m \times k$ columns; i.e., at each iteration the number of columns is increased by m. Hence, this method still has the drawback that as it requires a considerable level of memory it will fail if m is large or if the convergence is slow. The following convergence result can be proven.

Proposition 3.1. *[67] Let $A = V\Lambda V^{-1}$ be the eigenvalue decomposition of A. The k step low-rank Smith iterates satisfy :*

$$Trace(\mathcal{G}_c - \mathcal{P}_k^S) \leq \kappa(V)^2 \cdot m \cdot \rho(A)^{2k} \cdot Trace(\mathcal{G}_c),$$

where $\kappa(V)$ is the condition number of V. ■

Proof. We proceed as follows :

$$\begin{aligned} Trace(\mathcal{G}_c - \mathcal{P}_k^S) &= Trace\left(\sum_{i=k}^{\infty} A^i B B^* (A^*)^i\right) \\ &= \sum_{i=k}^{\infty} \|A^i B\|_F^2 \\ &\leq m \sum_{i=k}^{\infty} \|A^i B\|_2^2 \\ &\leq m \cdot \kappa(V)^2 \cdot \rho(A)^{2k} \cdot \sum_{i=0}^{\infty} \|A^i B\|_2^2 \\ &\leq m \cdot \kappa(V)^2 \cdot \rho(A)^{2k} \cdot Trace(\mathcal{G}_c). \end{aligned}$$

□

Let σ_i and $\hat{\sigma}_i$ denote the HSV resulting from the full-rank exact Gramians and the low-rank approximate Gramians respectively, i.e.,

$$\sigma_i^2 = \lambda_i(\mathcal{G}_c\mathcal{G}_o) \quad \text{and} \quad \hat{\sigma}_i^2 = \lambda_i(\mathcal{P}_k^S \mathcal{Q}_k^S).$$

Then, the following corollary holds.

Corollary 3.2. *[67] Let us define $\hat{N} = \min(m, p)$. Then*

$$\begin{aligned} \sum_{i=1}^{N} \sigma_i^2 - \sum_{i=1}^{k\hat{N}} \hat{\sigma}_i^2 \leq \kappa(V)^2 \rho(A)^{2k} \cdot \Big(&\kappa(V)^2 \hat{N} \rho(A)^{2k} \cdot Trace(\mathcal{G}_c) \cdot Trace(\mathcal{G}_o) + \\ &m \cdot Trace(\mathcal{G}_c) \sum_{i=0}^{k-1} \|CA^j\|_2^2 + p \cdot Trace(\mathcal{G}_o) \sum_{i=0}^{k-1} \|A^j B\|_2^2 \Big). \end{aligned}$$

■

Remark 3.3. One should notice that these error bounds critically depend on $\rho(A)$ and $\kappa(V)$. Hence when $\rho(A)$ is almost 1 and/or A is highly non-normal, these error bounds will be of little use.

3.3 Modified low-rank Smith method

As already mentioned above, the linear growth of the number of columns of $\mathcal{C}_k$ may cause storage problems and yield poor convergence. To avoid this kind of problems, we propose a modified low-rank Smith iteration so that the number n_k of columns in the low-rank square root factor does not increase unnecessarily at each step. The idea is not new, it has been already used in many others areas : image compression [40], information retrieval [20], but its application to model reduction was proposed first by Gugercin in [67].
Again, consider the Stein equation (3.1)

$$\mathcal{G}_c = A\mathcal{G}_c A^* + BB^*.$$

Then the k^{th} low-rank Smith iterate is simply given by (3.3) :

$$\mathcal{C}_k = \left[B \; AB \; A^2B \; \dots \; A^{k-1}B \right].$$

Hence, the approximate low-rank Gramian at the k^{th} step is

$$\mathcal{P}_k^S = \mathcal{C}_k \mathcal{C}_k^* = \sum_{i=0}^{k-1} A^i BB^* (A^i)^*.$$

Let the Short Singular Value Decomposition (SSVD) of $\mathcal{C}_k$ be $\mathcal{C}_k = U\Sigma V^*$. Then the SSVD of $\mathcal{P}_k^S$ is given by $\mathcal{P}_k^S = U\Sigma^2 U^*$. Hence, it is enough to store only U and Σ. In other words, $\hat{\mathcal{C}}_k = U\Sigma$ is also a low-rank square root factor of $\mathcal{P}_k^S$.
The idea is to replace the iterate $\mathcal{C}_k$ (or $\hat{\mathcal{C}}_k$) by its best rank n_k approximation S_k^{MLRS}, here the superscript Sm refers to *modified Smith*.
Indeed,

$$\mathcal{P}_k^{MLRS} \doteq S_k^{MLRS} (S_k^{MLRS})^* \quad \text{where} \quad S_k^{MLRS} = U(:, 1\!:\!n_k) \cdot \Sigma(1\!:\!n_k, 1\!:\!n_k)$$

is the best rank n_k approximation of $\hat{\mathcal{C}}_k \hat{\mathcal{C}}_k^*$ and $\mathcal{C}_k \mathcal{C}_k^*$, and so $\mathcal{P}_k^{MLRS}$ is the best rank n_k approximation of $\mathcal{P}_k^S$. However, instead of recomputing this approximation at each iteration, we propose an algorithm which updates it to include new data.
This algorithm is more general and can be applied in different fields. In fact, finding the best rank n_k approximation of $\mathcal{C}_k$ is equivalent to finding its dominant subspace of dimension n_k. And so the algorithm is a recursive calculation of the dominant n_k^{th} dimensional singular subspace of a matrix M. It uses knowledge of the singular subspace of the new system to construct transformation which isolates the dominant subspace of the updated matrix from the "noise" subspace, so that it may be truncated.
In the next chapter, we propose to study the general numerical problem. We present an elegant economical sequential procedure to compute S_k^{MLRS} and how to update it.

Chapter 4

Recursive calculation of dominant singular subspaces

In this chapter we show how to compute recursively low-rank approximations of the Gramians of LTI system. The algorithm computes recursively the dominant singular subspace of the controllability or observability matrices.

The numerical idea of this algorithm can also be applied to many other problems where one needs to compute the projector on the dominant subspace of a given data matrix M of dimension $N \times m$. The type of application we are thinking of here implies $N \gg m$, and for the sake of simplicity we will assume M to be real. In addition, we assume that the matrix M is produced incrementally, so all of the columns are not available simultaneously. Several applications have this property. For example, approximating a matrix M in which each column represents an image of a given sequence amounts to a Singular Value Decomposition-based compression [40]. Such an approximation is also used in the context of observation-based model reduction for dynamical systems. The so-called proper orthogonal decomposition (POD) approximation uses the dominant left space of a matrix M where a column consists of a time instance of the solution of an evolution equation, e.g., the flow field from a fluid dynamics simulation. Since these flow fields tend to be very large only a small number can be stored efficiently during the simulation, and therefore an incremental approach is useful [128]. The dominant space approximation is also used in text retrieval to encode document/term information and avoid certain types of semantic noise. The incremental form is required when documents are added or when the entire matrix is not available at one point in time and space [29]. Finally, this approach can be applied to compute the modified low-rank Smith approximation S_n^{MLRS}.

In each of these applications, one can interpret the columns of the matrix M as "data vectors" with some "energy" equal to their 2-norm. Finding the dominant space of dimension $n < \min(N, m)$ amounts to finding the n first columns of the matrix U in the singular value decomposition of M :

$$M = U\Sigma V^T, \quad U^T U = I_m, \quad VV^T = V^T V = I_m, \quad \Sigma = \mathrm{diag}\{\sigma_1, \dots \sigma_m\}, \tag{4.1}$$

and where the diagonal elements σ_i of Σ are non negative and non increasing. This decomposition in fact expresses that the orthogonal transformation V applied to the columns of M yields a new matrix $AV = U\Sigma$ with orthogonal columns of non-increasing norm. The "dominant" columns of this transformed matrix are obviously the n leading ones. A block version of this decomposition makes this more explicit :

$$M = U\Sigma V^T = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} \Sigma_1 & \\ & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1 & V_2 \end{bmatrix}^T, \tag{4.2}$$

where U_1 and V_1 have n columns and Σ_1 is $n \times n$. An orthogonal basis for the corresponding space is then given by U_1, which is also equal to $MV_1\Sigma_1^{-1}$. The cost of this decomposition including the construction of U is $14Nm^2 + O(m^3)$. For an additional $O(m^3)$ operations it is also possible to compute an orthogonal basis for the columns of V_1, which is required in several applications.

A cheaper procedure is to first perform a QR decomposition of M, followed by a singular value decomposition of the smaller matrix R [34] :

$$M = QR, \quad R = U\Sigma V^T. \tag{4.3}$$

From these equations it is easy to see that $MV = QU\Sigma$, and again this has orthogonal columns of non increasing norms. This decomposition costs typically $6Nm^2 + O(m^3)$ [63]. It is even more economical to use the normal equations (or covariance matrix) of M. Its eigenvalue decomposition gives

$$M^T M = V\Lambda V^T, \tag{4.4}$$

and comparing this with (4.1) shows that the same matrix V is constructed and that

$$(MV)^T(MV) = \Lambda = \Sigma^T\Sigma.$$

This algorithm requires Nm^2 operations to construct $M^T M$ and $Nmn + O(m^3)$ operations to obtain $U_1 = MV_1\Sigma_1^{-1}$. Unfortunately, using the covariance matrix is not recommended because it is more sensitive to rounding errors [63].

In this chapter we consider applications where N is huge, and where every column operation on M or on the basis U is not only costly in operations but also involves swapping data from the main memory, which will slow down the algorithm significantly. We present an algorithm that yields an approximate decomposition but requires only $8Nmn + 4Nmk + O(mn^3)$ operations (where k is the dimension of block that we add at each iteration) and also works recursively on the columns of M, i.e., the columns of M (or data vectors) can be produced recursively and M need not be stored in its entirety.

4.1 Recursive procedure

In this section we propose a recursive procedure to estimate the dominant subspace of a given matrix M using a sequential (and incremental) processing of the columns of M. Bounds for the accuracy of this decomposition are derived later.

The main idea of the algorithm is the use of a sliding window, in the sense that we track the n_i-dominant space of M with a sequence of windowed SVD's of finite dimension $N \times (n_i + k)$.

Let us define first some notations. We denote by M_i the current version of the "window", K_i the kept dominant part of this window after iteration i, N_i the added part to K_{i-1} at the iteration i to construct M_i, i.e.,

$$M_i = \left[K_{i-1}\ N_i \right], \tag{4.5}$$

and n_i the number of columns kept at each iteration. For now we will assume n_i to be a constant $n_i = n$ defined by user. We will see later how to automate the choice of n_i and adapt it dynamically.

Assume that at the beginning of step i a QR decomposition of $K_{i-1} \in \mathbb{R}^{N\times n}$ is available :

$$K_{i-1} = Q_{(i-1)}R_{(i-1)}. \tag{4.6}$$

By using a Gram-Schmidt procedure, it is possible to update this QR factorization by building on the current factorization. Given the following :

$$R_N = Q_{(i-1)}^* N_i,$$

$$(I - Q_{(i-1)}Q_{(i-1)}^*)N_i = N_i - Q_{(i-1)}R_N = Q_\perp R_\perp, \tag{4.7}$$

the updated QR factorization is given by the following partitioning :

$$M_i = \left[Q_{(i-1)}R_{(i-1)}\ N_i \right] = \left[Q_{(i-1)}\ Q_\perp \right] \begin{bmatrix} R_{(i-1)} & R_N \\ & R_\perp \end{bmatrix}$$
$$= \hat{Q}\hat{R}.$$

The construction of $(N_i - Q_{(i-1)}R_N)$ requires $4Nnk$ operations, while its QR factorization (4.7) requires $4Nk^2$. This results in a total cost of $4Nk(n+k)$ operations per step.

In order to truncate the information associated with the residual subspace denoted $Q_{(i)_{res}}$, it is necessary to isolate it from the dominant subspace $Q_{(i)}$ in $\hat{Q}$.
This can be done by applying transformations G_u and G_v to both sides of $\hat{R}$ to decouple the dominant and residual subspaces as follows :

$$G_u^* \hat{R} G_v = G_u^* \begin{bmatrix} R_{(i-1)} & R_N \\ 0 & R_\perp \end{bmatrix} G_v = \begin{bmatrix} R_{(i)} & 0 \\ 0 & R_{(i)_{res}} \end{bmatrix}. \tag{4.8}$$

Now, let us consider the SVD of $\hat{R}$:

$$\hat{R} = \begin{bmatrix} \hat{U}_1 \ \hat{U}_2 \end{bmatrix} \begin{bmatrix} \hat{\Sigma}_1 & \\ & \hat{\Sigma}_2 \end{bmatrix} \begin{bmatrix} \hat{V}_1 \ \hat{V}_2 \end{bmatrix},$$

where $\hat{\Sigma}_1 \in \mathbb{R}^{n \times n}$ contains the dominant singular values and $\hat{\Sigma}_2 \in \mathbb{R}^{k \times k}$ contains the dismissed singular values.
G_u is constructed to transform $\hat{U}_2$ to $\begin{bmatrix} 0 \\ I_k \end{bmatrix}$. Applying G_u to $\hat{R}$ will destroy its upper triangular form, and so G_v is constructed to restore $G_u^* \hat{R}$ to upper triangular form.
We therefore have the updating QR decomposition :

$$\begin{aligned} M_i G_v = Q_{up} R_{up} = & \quad (\hat{Q} G_u) \quad \cdot \quad (G_u^* \hat{R} G_v) \\ & = \begin{bmatrix} Q_{(i)} \ Q_{(i)_{res}} \end{bmatrix} \cdot \begin{bmatrix} R_{(i)} & 0 \\ 0 & R_{(i)_{res}} \end{bmatrix}, \end{aligned} \tag{4.9}$$

and since R_{up} has the required block form (4.1) we have found a basis for the dominant n dimensional subspace of M_i in the form of the first n columns of Q_{up} i.e., $Q_{(i)}$. Furthermore, the new K_i is obtained as follows :

$$K_i = Q_{(i)} R_{(i)}.$$

Both matrices G_u and G_v can be constructed as a product of $\frac{k(2n+k-1)}{2}$ 2×2 Givens transformations, allowing an elegant update of $\hat{R}$ using only $O((nk)^2)$ operations (see Appendix 8). But the costly part of our algorithm is the update of $\hat{Q}$, and hence it is preferable to choose G_u to be a Householder transformation. When retriangularizing $G_u^* \hat{R}$ one then needs to perform again a QR factorization, which requires $2(n+k)^3$ operations (a Gram-Schmidt on a $(n+k) \times (n+k)$ matrix), but since $n, k < m \ll N$, this is of no concern. The cost of the update of $\hat{Q}$ to Q_{up} is that of a Householder transformation applied to an $N \times (n+k)$ matrix (but where only the n first columns are computed) and is thus $4Nnk$ operations. The column vectors of the matrix $\hat{U}_2$ can be computed with a few steps of inverse iteration, or with a shifted inverse iteration. The cost of this calculation as well as the update of $\hat{R}$ is thus $8Nnk + 4Nk^2 + O((n+k)^3)$ operations per iteration and hence negligible with respect to the update of $\hat{Q}$. A more involved technique uses modified Givens transformations since their complexity is the same as that of Householder transformations for the product $\hat{Q} G_u$, and is of the order of $10Nnk + 4Nk^2$ for forming the product $G_u^* \hat{R} G_v$. Unfortunately, this requires storing and updating additional diagonal scaling matrices, which typically hurt the performance of codes used for parallel machines. We refer to [13] for a discussion on the Householder and Givens transformations for the block case.
Table 4.1 shows the complexity of each step and the total complexity for the block case, i.e., where we add $k > 1$ new vectors at each step. We also show the total complexity for the single vector case where at each step we add only one vector (i.e., $k = 1$). We recall that these complexities are the cost of our algorithm, for both techniques : Givens and Householder, applied to a $N \times m$ matrix, where the number of steps is given by τ, $m = k\tau$.

Algorithm	Givens	Householder
Cost per step ($k>1$)	$10Nnk+4Nk^2+6nk^2i$	$8Nnk+4Nk^2+2n(n+k)ik$
Total cost ($k>1$)	$10Nn\tau+4Nk\tau+3nk^2\tau(\tau+1)$	$8Nn\tau+4Nk\tau+n(n+k)k\tau(\tau+1)$
Total cost ($k=1$)	$10Nmn+O(mn^3)$	$8Nmn+O(mn^3)$

Table 4.1. Two-sided complexity for Givens and Householder [13].

Algorithm 5 shows the formal algorithm of this recursive procedure.

Algorithm 5 The Modified Low-Rank Smith algorithm (MLRS).

Given n and k s.t. $m = n + k\tau$, we initialize K_0 by

$$K_0 = M(:, 1:n),$$

and let define $k_1 = n + k$;
for $i = 1 : \tau$ **do**
  $N_i = M(:, n + (i-1)k + 1 : n + ik)$;
  $M_i = \left[\, K_{i-1} \middle| N_i \,\right]$;
    % We expand K_{i-1} by appending k columns N_i.
  $[Q, R] = qr(M_i, 0)$;
    %We compute an economical QR decomposition of M_i;
  $[U, S, V] = svd(R, 0)$;
    %We compute the SSVD of the small upper triangular matrix R;
  for $j = n + k : -1 : n + 1$ **do**
    for $l = 1 : k_1 - 1$ **do**
      $Gu = givens(U(i+1, j), U(i, j))$;
      $U(i : i+1, :) = Gu' * U(i : i+1, :)$;
      $R(i : i+1, :) = Gu' * R(i : i+1, :)$;
      $Gv = givens(R(i+1, j-n+2), R(i+1, j-n+1))$;
      $R(:, j-n+1 : j-n+2) = R(:, j-n+1 : j-n+2) * Gv$;
      $Q(:, i : i+1) = Q(:, i : i+1) * Gu$;
    end for
    $k_1 = k_1 - 1$;
  end for
  $K_i = Q(:, 1:n) * R(1:n, 1:n)$;
    % We update K_i using a contraction (deleting k columns).
end for

The algorithm thus computes at each step a decomposition that "deflates" the smallest singular vectors of the current $N \times (n+k)$ matrix M_i and then appends to it the columns of N_i. All columns of M therefore are passed through once and compared with the current best estimate of this dominant subspace. At first sight this is a very heuristic algorithm but in Section 4.3 we show that quite good bounds can be obtained for the quality of this basis (see also [13]).

4.2 Updating a two-sided decomposition

The algorithm above yields at step i an approximation $Q_{(i)}$ of the dominant left singular subspace of $M(:, 1 : n + ik)$, but in several applications it makes sense to update simultaneously an approximation of the corresponding right singular subspace of this matrix. This can be done with little extra cost.

We start from the notation introduced in (4.6), which we rewrite as

$$M(:, 1 : n + k)V_{(1)} = Q_{(1)}R_{(1)}, \tag{4.10}$$

where $V_{(1)} = I_{n+k}G_v \begin{bmatrix} I_n \\ \mathbf{0} \end{bmatrix} \in \mathbb{R}^{n+k\times n}$. We show by induction that at each step i we have a decomposition

$$M(:, 1 : n + ik)V_{(i)} = K_i = Q_{(i)}R_{(i)}, \tag{4.11}$$

where $V_{(i)} \in \mathbb{R}^{n+ik\times n}$ satisfies $V_{(i)}^T V_{(i)} = I_n$.
From (4.10) it is obvious that this holds for $i = 1$. For the induction step we start by assuming that it holds for $i - 1$:

$$M(:, 1 : n + (i - 1)k)V_{(i-1)} = K_i = Q_{(i-1)}R_{(i-1)}.$$

We then append the block N_i to $M(:, n + (i - 1)k)$ to get $M(:, n + ik)$ and obviously

$$M(:, 1 : n + ik) \begin{bmatrix} V_{(i-1)} & 0 \\ 0 & I_k \end{bmatrix} = \begin{bmatrix} Q_{(i-1)}R_{(i-1)} & N_i \end{bmatrix}. \tag{4.12}$$

Now use (4.5–4.9) to update this to

$$\begin{aligned} M(:, 1 : n + ik) \begin{bmatrix} V_{(i-1)} & 0 \\ 0 & I_k \end{bmatrix} G_v &= \begin{bmatrix} Q_{(i)}R_{(i)} & Q_{(i)_{res}}R_{(i)_{res}} \end{bmatrix} \\ &= \begin{bmatrix} K_i & Q_{(i)_{res}}R_{(i)_{res}} \end{bmatrix}. \end{aligned} \tag{4.13}$$

Taking the first n columns of both sides of this equation yields (4.11) with

$$V_{(i)} = \begin{bmatrix} V_{(i-1)} & 0 \\ 0 & I_k \end{bmatrix} G_v \begin{bmatrix} I_n \\ 0 \end{bmatrix} \in \mathbb{R}^{(n+ik)\times n}, \tag{4.14}$$

which obviously satisfies $V_{(i)}^* V_{(i)} = I_n$. The additional work for updating the approximation $V_{(i)}$ is just the multiplication (4.14), which requires $2n(n + k)ik$ flops per iteration if G_v is constructed via Householder transformations and $6nk^2i$ flops per iteration if we use the Givens rotations, and hence leads, respectively, to a total of

$$\sum_{i=1}^{\tau} 2n(n + k)ik = 2nk(n + k)\sum_{i=1}^{\tau} i = nk(n + k)\tau(\tau + 1),$$

and

$$\sum_{i=1}^{\tau} 6nk^2i = 6nk^2\sum_{i=1}^{\tau} i = 3nk^2\tau(\tau + 1),$$

additional flops for the full decomposition[1]. This additional work can be neglected if $N \gg k, n$.

We terminate this section by writing a decomposition for the matrix $M(:, 1 : n + ik)$ (for any $i \geq 1$) if we would not delete the "residual" block at each step. There exists an orthogonal matrix $V_i \in \mathbb{R}^{(ik+n)\times(ik+n)}$ embedding $V_{(i)}$:

$$V_i = \begin{bmatrix} V_{(i)} & V_{(i)}^{\perp} \end{bmatrix}.$$

Choosing appropriate basis vectors for $V_{(i)}^{\perp}$, we obtain a decomposition of the type

$$M(:, 1 : n + ik)V_i = \begin{bmatrix} Q_{(i)}R_{(i)} & Q_{(i)_{res}}R_{(i)_{res}} & \dots & Q_{(1)_{res}}R_{(1)_{res}} \end{bmatrix}. \tag{4.15}$$

[1] Recall that $m = n + \tau k$.

From this we obtain the additive decomposition

$$M(:,1:n+ik) = \underbrace{Q_{(i)}R_{(i)}}_{K_i} V_{(i)}^* + \left[\, Q_{(i)_{res}}R_{(i)_{res}} \,\ldots\, Q_{(1)_{res}}R_{(1)_{res}} \,\right] V_{(i)}^{\perp T} \tag{4.16}$$

which will be used later on to derive error bounds.

4.3 Accuracy bounds

It is clear that after the first step we obtain a decomposition

$$M(:,1:n+k)G_v^* = \left[\, Q_{(1)}\; Q_{(1)_{res}} \,\right] \cdot \begin{bmatrix} R_{(1)} & 0 \\ 0 & R_{(1)_{res}} \end{bmatrix}. \tag{4.17}$$

Let σ_i, $i = 1,\ldots,m$, be the singular values of M, $\hat{\sigma}_j^{(1)}$, $j = 1,\ldots,n$, those of $R_{(1)}$ and thus they are also the singular values of K_1, and $\mu_j^{(1)}$, $j = 1,\ldots,k$ those of $R_{(1)_{res}}$. Then according to the above decomposition, $M(:,1:n+k)$ has singular values

$$\hat{\sigma}_1^{(1)},\ldots,\hat{\sigma}_n^{(1)},\mu_1^{(1)},\ldots,\mu_k^{(1)}.$$

But since this is a submatrix of M obtained by deleting a number of columns, we have the inequalities [63] :

$$\hat{\sigma}_1^{(1)} \le \sigma_1, \quad \ldots, \quad \hat{\sigma}_n^{(1)} \le \sigma_n, \quad \mu_1^{(1)} \le \sigma_{n+1}, \quad \ldots, \quad \mu_k^{(1)} \le \sigma_{n+k}. \tag{4.18}$$

Similarly one easily shows that each intermediate matrix (4.9)

$$M_iG_v = \left[\, Q_{(i)}\; Q_{(i)_{res}} \,\right] \cdot \begin{bmatrix} R_{(i)} & 0 \\ 0 & R_{(i)_{res}} \end{bmatrix} \tag{4.19}$$

with singular values

$$\hat{\sigma}_1^{(i)},\ldots,\hat{\sigma}_n^{(i)},\mu_1^{(i)},\ldots,\mu_k^{(i)}$$

is also orthogonally equivalent to a submatrix of M. Therefore we have in general

$$\hat{\sigma}_1^{(i)} \le \sigma_1, \quad \ldots, \quad \hat{\sigma}_n^{(i)} \le \sigma_n, \quad \mu_1^{(i)} \le \sigma_{n+1}, \quad \ldots, \quad \mu_k^{(i)} \le \sigma_{n+k}. \tag{4.20}$$

Finally, since the matrix

$$\begin{aligned} \left[\, K_{i-1}\; N_i \,\right] &= \left[\, Q_{(i-1)}R_{(i-1)}\; Q_{(i-1)}R_N + Q_\perp R_\perp \,\right] \\ &= \left[\, Q_{(i)}\; Q_{(i)_{res}} \,\right] \begin{bmatrix} R_{(i)} & 0 \\ 0 & R_{(i)_{res}} \end{bmatrix} G_v^* \\ &= \left[\, K_i\; Q_{(i)_{res}}R_{(i)_{res}} \,\right] G_v^* \end{aligned} \tag{4.21}$$

has singular values $\hat{\sigma}_1^{(i)},\ldots,\hat{\sigma}_n^{(i)},\mu_1^{(i)},\ldots,\mu_k^{(i)}$, and $Q_{(i-1)}R_{(i-1)}$ is its submatrix, we have the inequalities

$$\hat{\sigma}_1^{(i-1)} \le \hat{\sigma}_1^{(i)}, \quad \ldots, \quad \hat{\sigma}_n^{(i-1)} \le \hat{\sigma}_n^{(i)}. \tag{4.22}$$

All this says that the singular values $\mu_j^{(i)}$, $j = 1,\ldots,k$, that are dismissed at each step are all smaller than σ_{n+1} and that the singular values $\hat{\sigma}_j^{(i)}$, $j = 1,\ldots,n$, that are updated increase monotonically towards the first n singular values of M. To obtain bounds at the end of the iterative procedure we need to relate M to the computed quantities. For this, we point out that there exists an orthogonal column transformation V_τ (4.15) which relates M and the intermediate results of the recursive algorithm, where τ is the number of iterations :

$$MV_\tau = \left[K_\tau \; Q_{(\tau)_{res}} R_{(\tau)_{res}} \; \ldots \; Q_{(1)_{res}} R_{(1)_{res}} \right]. \tag{4.23}$$

The transformation V_τ indeed consists of all the smaller transformations G_v and appropriately chosen permutations to obtain (4.23). Using the singular value decomposition of $R_{(\tau)}$,

$$R_{(\tau)} = \hat{U}_\tau \Sigma \hat{V}_\tau^*,$$

one then constructs orthogonal transformations so that

$$MV_\tau \begin{bmatrix} \hat{V}_\tau & 0 \\ 0 & I \end{bmatrix} = \left[Q_{(\tau)} \hat{U}_\tau \; Q_{(\tau)}^\perp \right] \begin{bmatrix} \hat{\Sigma} & R_1^e \\ 0 & R_2^e \end{bmatrix}, \tag{4.24}$$

where $Q_{(\tau)}^\perp$ is orthogonal to $Q_{(\tau)}$ and where the columns of

$$R^e \doteq \begin{bmatrix} R_1^e \\ R_2^e \end{bmatrix}$$

have 2-norms $\mu_j^{(i)}$. The Frobenius norm of this submatrix is therefore equal to

$$\left\| \left[\mu_1^{(1)}, \ldots, \mu_k^{(1)}, \mu_1^{(2)}, \ldots, \mu_k^{(2)}, \ldots, \mu_1^{(\tau)}, \ldots, \mu_k^{(\tau)} \right] \right\|_2.$$

From (4.24) one already finds a bound for the accuracy of the computed singular values. The singular values of M are also those of $\hat{M} \doteq \begin{bmatrix} \hat{\Sigma} & R_1^e \\ 0 & R_2^e \end{bmatrix}$. Applying the Wielandt-Hoffman theorem for singular values to this yields [63] :

$$\sum_{i=1}^{n} (\sigma_i - \hat{\sigma}_i^{(\tau)})^2 \leq \|R^e\|_F^2 = \sum_{i,j} (\mu_j^{(i)})^2 \leq (m-n) \cdot \sigma_{n+1}^2. \tag{4.25}$$

If we know the singular values have a considerable gap

$$\gamma \doteq \sigma_n - \sigma_{n+1}, \tag{4.26}$$

then this bound says that the n largest singular values are well approximated. If γ is large, the space spanned by the corresponding singular vectors is also insensitive to perturbations. Moreover, one can improve on the bounds for the singular value perturbations provided by the Wielandt-Hoffman theorem. To analyze this in more detail we use the following theorem proven in [116].

Theorem 4.1.

Let $\hat{H}$ and E be square Hermitian matrices partitioned as

$$\hat{H} = \begin{bmatrix} \hat{H}_{1,1} & 0 \\ 0 & \hat{H}_{2,2} \end{bmatrix}, \quad E = \begin{bmatrix} E_{1,1} & E_{1,2} \\ E_{2,1} & E_{2,2} \end{bmatrix},$$

and define $\epsilon = \|E_{1,2}\|_2$ and $\delta = \min |\lambda(\hat{H}_{1,1}) - \lambda(\hat{H}_{2,2})| - \|E_{1,1}\|_2 - \|E_{2,2}\|_2$.
If $\delta > 2\epsilon$, then there exists a unitary matrix X of the form

$$X = \begin{bmatrix} I_n & -P^* \\ P & I_{N-n} \end{bmatrix} \begin{bmatrix} (I + P^*P)^{-1/2} & 0 \\ 0 & (I + PP^*)^{-1/2} \end{bmatrix}$$

such that

$$H \doteq X^*(\hat{H} + E)X = \begin{bmatrix} H_{1,1} & 0 \\ 0 & H_{2,2} \end{bmatrix},$$

where $\|P\|_2 < 2\epsilon/\delta$. ■

This theorem is used to estimate the accuracy of both the left and right dominant subspaces of M as follows. Suppose

$$\hat{H}_u = \begin{bmatrix} \hat{\Sigma}^2 & 0 \\ 0 & 0 \end{bmatrix} \tag{4.27}$$

is the current "approximation" of the eigenvalue decomposition of

$$H_u \doteq \hat{M}\hat{M}^* = \begin{bmatrix} \hat{\Sigma}^2 & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} R_1^e \\ R_2^e \end{bmatrix} \begin{bmatrix} (R_1^e)^* & (R_2^e)^* \end{bmatrix}. \tag{4.28}$$

The left dominant "singular" subspace of $\hat{M}$ is also the dominant eigensubspace of H_u. The dominant eigensubspace of the nearby matrix $\hat{H}_u$ is clearly $\mathrm{Im}\begin{bmatrix} I_n \\ 0 \end{bmatrix}$ and the corresponding eigenvalues are the diagonal elements $\hat{\sigma}_1^{(\tau)}, \ldots, \hat{\sigma}_n^{(\tau)}$ of $\hat{\Sigma}^2$. But due to the perturbations R_1^e and R_2^e these are incorrect. After transforming $\hat{M}\hat{M}^*$ to $X_u^* \hat{M}\hat{M}^* X_u$ we obtain its true eigenvalues (i.e., the squared singular values of $\hat{M}$) in the matrix $H_{1,1}$ and the true dominant subspace as $\mathrm{Im}\begin{bmatrix} I_n \\ P_u \end{bmatrix}$. The norm of P_u is a measure for the angular rotation of this subspace and it is bounded by $2\epsilon_u/\delta_u$. The largest canonical angle θ_n between the spaces $\mathrm{Im}\begin{bmatrix} I_n \\ 0 \end{bmatrix}$ and $\mathrm{Im}\begin{bmatrix} I_n \\ P_u \end{bmatrix}$ in fact satisfies [116] :

$$\cos\theta_n = \frac{1}{\sqrt{1+\|P_u\|^2}}, \quad \sin\theta_n = \frac{\|P_u\|}{\sqrt{1+\|P_u\|^2}}, \quad \tan\theta_n = \|P_u\|$$

and measures the "rotation" of the dominant subspace with respect to its approximation.
Clearly here $\epsilon_u = \|R_1^e (R_2^e)^*\|_2$ and $\delta_u = (\hat{\sigma}_n^{(\tau)})^2 - \|R_1^e\|_2^2 - \|R_2^e\|_2^2$. Notice that $\|R^e\|_F^2 = \sum_{i,j} (\mu_j^{(i)})^2$ and that we actually compute these values during our recursive calculations. It would therefore be convenient to bound $2\epsilon_u/\delta_u$ in terms of these "discarded" singular values $\mu_j^{(i)}$. One easily derives the bounds

$$\|R_1^e (R_2^e)^*\|_2 \le \frac{1}{2} \left\| \begin{bmatrix} R_1^e \\ R_2^e \end{bmatrix} \begin{bmatrix} (R_1^e)^* & (R_2^e)^* \end{bmatrix} \right\|_2 = \frac{1}{2} \left\| \underbrace{\begin{bmatrix} R_1^e \\ R_2^e \end{bmatrix}}_{R^e} \right\|_2^2$$

and

$$\|R^e\|_2^2 \le \|R_1^e (R_2^e)^*\|_2 + \|R_2^e (R_2^e)^*\|_2 = \|(R_1^e)^* R_1^e\|_2 + \|(R_2^e)^* R_2^e\|_2 \le 2\|R^e\|_2^2.$$

Defining

$$\mu \doteq \left\| \begin{bmatrix} R_1^e \\ R_2^e \end{bmatrix} \right\|_2 \tag{4.29}$$

we then have

$$\epsilon_u \le \frac{\mu^2}{2}, \quad (\hat{\sigma}_n^{(\tau)})^2 - \mu^2 \ge \delta_u \ge (\hat{\sigma}_n^{(\tau)})^2 - 2\mu^2, \tag{4.30}$$

and provided that $\hat{\sigma}_n^{(\tau)} \ge \sqrt{3}\mu$ we obtain

$$\delta_u \ge 2\epsilon_u \Rightarrow \|P_u\|_2 \le \frac{2\epsilon_u}{\delta_u}.$$

For the right dominant singular subspace of $\hat{M}$ we must consider

$$H_v \doteq \hat{M}^* \hat{M} = \begin{bmatrix} \hat{\Sigma}^2 & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & \hat{\Sigma} R_1^e \\ (R_1^e)^* \hat{\Sigma} & (R_1^e)^* R_1^e + (R_2^e)^* R_2^e \end{bmatrix}. \tag{4.31}$$

For the quantities ϵ_v and δ_v corresponding to Theorem 4.1, we find :

$$\epsilon_v \doteq \|\hat{\Sigma} R_1^e\|_2 \leq \mu \|M\|_2, \quad \delta_v \doteq \min |\lambda(\hat{\Sigma}^2)| - \|R^e\|_2^2 = (\hat{\sigma}_n^{(\tau)})^2 - \mu^2.$$

Provided that $(\hat{\sigma}_n^{(\tau)})^2 \geq \frac{16}{7} \mu \|M\|_2$ we obtain

$$\delta_v \geq 2\epsilon_v \Rightarrow \|P_v\|_2 \leq 2\frac{\epsilon_v}{\delta_v}.$$

Applying the same reasoning as above we denote the true dominant subspace as $\mathrm{Im} \begin{bmatrix} I_{n\tau} \\ P_v \end{bmatrix}$. The norm of P_v is then a measure for the angular rotation of this subspace, and it is bounded by $2\epsilon_v/\delta_v$. The corresponding largest canonical angle ϕ_k satisfies again [116] :

$$\cos \phi_n = \frac{1}{\sqrt{1 + \|P_v\|^2}}, \quad \sin \phi_n = \frac{\|P_v\|_2}{\sqrt{1 + \|P_v\|^2}}, \quad \tan \phi_n = \|P_v\|_2$$

and measures the "rotation" of the right dominant singular subspace with respect to its approximation. We summarize this discussion in the following theorem.

Theorem 4.2.
Let

$$\bar{M} = \begin{bmatrix} \hat{\Sigma} & 0 \\ 0 & 0 \end{bmatrix}, \quad \hat{M} = \begin{bmatrix} \hat{\Sigma} & R_1^e \\ 0 & R_2^e \end{bmatrix}, \quad \mu \doteq \|R^e\|_2.$$

Then the angles θ_n and ϕ_n between the n-dimensional left and right singular subspaces of $\hat{M}$ and $\bar{M}$, respectively, satisfy the bounds

$$\tan \theta_n < \frac{\mu^2}{((\sigma_n^{(\tau)})^2 - 2\mu^2)} \text{ if } \mu < \frac{\sigma_n^{(\tau)}}{\sqrt{3}}$$

and

$$\tan \phi_n < \frac{\mu \|\hat{M}\|_2}{((\sigma_n^{(\tau)})^2 - \mu^2)} \text{ if } \mu < \frac{7(\sigma_n^{(\tau)})^2}{16\|M\|_2}.$$

These are also the angles of the left and right singular subspaces of $Q_{(i)} R_{(i)} V_{(i)}^*$ and M. ■

Unfortunately, we do not compute the matrices R_1^e and R_2^e, and so we have to estimate μ. Bounding μ^2 in terms of the Frobenius norm :

$$\mu^2 \leq \sum_{i,j} (\mu_j^{(i)})^2$$

would yield serious overestimates since δ may become negative. Therefore we have to make some simplifying assumptions. The i-th block of R^e at step i of the recursive calculation contains what could be considered as "residual noise vectors", and we assume therefore that they are randomly distributed. It is shown in [60] that an $(m-n) \times m$ matrix M with elements chosen independently from a standard Gaussian distribution has column norms tending to $\sqrt{m}$ and a spectral norm $\|M\|_2$ tending to $\sqrt{m}(1 + \sqrt{(m-n)/m})$ as m becomes large. If our matrix R^e has equal column norms (hence equal to $\max_{i,j} \mu_j^{(i)}$ rather than $\sqrt{N}$) we then obtain the approximation :

$$\max_{i,j} \mu_j^{(i)} \leq \mu \leq c. \max_{i,j} \mu_j^{(i)}, \quad c \approx \frac{1+\sqrt{(m-n)}}{m}.$$

On the other hand, if the columns are of very different norm, one gets closer to the lower bound since the number of relevant columns entering the above analysis, becomes smaller than $(m-n)$, and thus c tends to 1. We will simply use $\hat{\mu} = \max_{i,j} \mu_j^{(i)}$ and $\hat{\sigma}_1^{(\tau)}$, respectively, estimates of μ and $\|M\|_2$, which leads to the following approximations for our bounds :

$$\hat{\epsilon}_u \approx \hat{\mu}^2/2, \quad \hat{\delta}_u \approx (\hat{\sigma}_n^{(\tau)})^2 - \hat{\mu}^2, \quad \hat{\epsilon}_v \approx \hat{\mu}\hat{\sigma}_1^{(\tau)}, \quad \hat{\delta}_v \approx (\hat{\sigma}_n^{(\tau)})^2 - \hat{\mu}^2.$$

Notice that these approximations have the advantage that $\hat{\delta}_u$ and $\hat{\delta}_v$ will always be positive since $\sigma_n^{(\tau)} \geq \sigma_{n+1}^{(i)} = \mu_1^{(i)}$. The resulting estimates for the norm of P_u and P_v then become

$$\|P_u\|_2 \approx \tan\hat{\theta}_n \doteq 2\frac{\hat{\epsilon}_u}{\hat{\delta}_u} = \frac{\hat{\mu}^2}{(\hat{\sigma}_n^{(\tau)})^2 - \hat{\mu}^2}, \tag{4.32}$$

$$\|P_v\|_2 \approx \tan\hat{\phi}_n \doteq 2\frac{\hat{\epsilon}_u}{\hat{\delta}_u} = \frac{\hat{\mu}\hat{\sigma}_1^{(\tau)}}{(\hat{\sigma}_n^{(\tau)})^2 - \hat{\mu}^2}. \tag{4.33}$$

It is possible to estimate the quality of the computed singular values using a simpler analysis. From Theorem 4.1 it follows that

$$\Xi \begin{bmatrix} I + P^* \end{bmatrix} \left(\begin{bmatrix} \hat{\Sigma}^2 & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} K_1 \\ K_2 \end{bmatrix} \begin{bmatrix} K_1^* & K_2^* \end{bmatrix} \right) \begin{bmatrix} I \\ P \end{bmatrix} \Xi = H_{1,1}, \tag{4.34}$$

where

$$\Xi = (I + P^*P)^{-\frac{1}{2}}, \quad \Xi = \Xi^* \leq I.$$

This yields the residual equation

$$H_{1,1} - \Xi\hat{\Sigma}^2\Xi = R \doteq \Xi \begin{bmatrix} I & P^* \end{bmatrix} \begin{bmatrix} K_1 \\ K_2 \end{bmatrix} \begin{bmatrix} K_1^* & K_2^* \end{bmatrix} \begin{bmatrix} I \\ P \end{bmatrix} \Xi,$$

and since

$$\Xi\hat{\Sigma}^2\Xi \leq \hat{\Sigma}^2$$

we have

$$H_{1,1} - \hat{\Sigma}^2 \leq H_{1,1} - \Xi\hat{\Sigma}^2\Xi = R.$$

But

$$\|R\|_2 = \left\| \begin{bmatrix} R_1^e \\ R_2^e \end{bmatrix} \right\|_2^2 = \mu^2,$$

from which we obtain the strict bound

$$|\sigma_i^2 - (\hat{\sigma}_i^{(\tau)})^2| \leq \|H_{1,1} - \hat{\Sigma}^2\|_2 \leq \mu^2.$$

This analysis is very simple and does not take into account any information about P, which can be used to improve the bound. Instead, we replace μ by its estimate $\hat{\mu}$, which yields

$$|\sigma_i - \hat{\sigma}_i^{(\tau)}| \approx \frac{\hat{\mu}^2}{(\sigma_i + \hat{\sigma}_i^{(\tau)})} \leq \frac{\hat{\mu}^2}{2\hat{\sigma}_i^{(\tau)}}. \tag{4.35}$$

We point out that all of the estimates are quadratic in $\hat{\mu}$, which should give very accurate results if $\hat{\mu} \ll \hat{\sigma}_i^{(\tau)}$. This is the case if the gap γ at the n-th singular value is large, and the quality of the estimate should be expected to deteriorate when this gap becomes small. We illustrate the quality of these bounds in the examples of the next section.

Remark 4.3.

If M has rank n, then this approach produces an exact decomposition since each submatrix M_i has rank less than or equal to n and hence $\mu_i = 0$ at each step.

4.4 The choice of n_i

So far we have only considered the case where the number of kept vectors n_i at each iteration is constant and fixed from the beginning by the user.
But actually, if one wants to choose a convenient value for n_i one has to do an explicit thorough analysis of the whole matrix involved and strive for some sort of singular values ranking. For large-scale dimension this pre-treatment is prohibitive.

The current situation is that we can choose dynamically the number of vectors kept during the iterations of the algorithm (i.e., n_i is variable). This is very cheap as we already pass through the whole matrix with a sliding window which sorts locally the singular values. And so one can adapt n_i as soon as the information "unveiled" by the sliding window is interesting.
Here, one can adopt many strategies using some ad-hoc specification. e.g.

- *Absolute tolerance strategy*.
  In this case, one has to predefine a tolerance value ς_a and ask the algorithm to neglect all singular values which are smaller than this tolerance value, i.e.,

$$n_i = \min j, \quad \text{where} \quad \sigma_j^{(i)} < \varsigma_a.$$

- *Relative tolerance strategy*.

  This strategy is more dynamic and more interesting. First, one has to choose a "reasonable" n_1 in the beginning, reasonable in the sense that we choose to have at least a rank n_1-approximation. One can take for example $n_1 = 1$ if we want to automate the whole process. And let ς_r be a pre-specified tolerance value. At each iteration $i = 1 : \ldots$ we apply our algorithm and we check for all computed singular values $\sigma_j^{(i)}$, $j = 1 : n_i + m$, where $\sigma_j^{(i)} = \mu_{j-n_i}^{(i)}$ for $j = n_i + 1 : n_i + m$ the quotient :

$$\frac{\sigma_1^{(i)}}{\sigma_j^{(i)}} > \varsigma_r, \quad \text{for} \quad j = 1 : n_i + m.$$

  The first j for which we will have

$$\frac{\sigma_1^{(i)}}{\sigma_j^{(i)}} < \varsigma_r,$$

  is compared to n_i, if this j is smaller than n_i we take the next n_{i+1} equal to n_i (i.e., $n_{i+1} = n_i$), if not we take $n_{i+1} = j$, and so on.
  Of course, the pre-specification of $\varsigma_\bullet$ will be crucial and actually this choice is heuristic. Actually this tolerances has to be chosen by trial and error.

- Another strategy can be adopted for the choice of n_i. It is based on the remark that the quality of the approximation depends on the gap between the retained values and the neglected ones. So one can detect the gaps between singular values in each window, and adapt n as for the relative tolerance strategy.

Remark 4.4.

In the strategies above, because $n_1 \leq n_2 \leq \ldots \leq n_\tau$, if one keeps in memory all values of n_i, we can choose at the end the low-rank approximation only from the n_τ-rank approximation, which will embed all other n_i-rank approximations.

4.5 Numerical tests of the approximation

We generated random matrices M of dimension $N = 1000$ by $m = 50$ and attempted to track the $n = 5$ dominant singular values and vectors. At every step we keep at most $n+1 = 6$ vectors in our basis. We thus update to a subspace of dimension 6 and then deflate the smallest singular value to fall back to a space of dimension 5 at each step.

In Figures 4.1-4.4, the true singular values σ_i $(i = 1, \ldots, m)$ are represented by the solid line, the approximations $\sigma_i^{(m-n)}$ of the $i = 1, \ldots, n$ leading singular values are the asterisks and the dismissed singular values μ_i $(i = n+1, \ldots, m)$ are the circles. Four different gaps are used to illustrate the trend of a larger gap γ (4.26)

$$\gamma \doteq \sigma_n - \sigma_{n+1},$$

improving the quality of the approximations. Each figure is accompanied by a table listing the singular values σ_i, their approximations $\hat{\sigma}_i^{(m-n)}$, the corresponding errors $|\sigma_i - \hat{\sigma}_i^{(m-n)}|$ and their estimate

$$\hat{\mu}^2/(2\hat{\sigma}_i^{(m-n)}),$$

and finally the cosines of the canonical angles $\cos\theta_i$ and $\cos\phi_i$, the smallest of which indicates the rotation of the dominant left and right singular subspaces versus their approximation, and the estimated angles $\cos\hat{\theta}_n$ and $\cos\hat{\phi}_n$. We also give the true value of μ, its estimate $\hat{\mu}$, and finally the $(n+1)$-st singular value.

From these examples it appears that the method works reasonably well. It should be pointed out that Theorem 4.2 applies only to the last example and that the estimates are very good. Nevertheless the estimates are still acceptable even when the conditions of this theorem do not apply, as is shown by the first example, which has virtually *no* gap! Notice that $\mu/\hat{\mu}$ remains smaller than 2, as suggested by the statistical arguments of Section 4.3. We also analyzed intermediate values of γ, which confirmed the remarks made above.

LEGEND FOR FIGURES 4.1–4.4 :
— true sv's $\sigma_i(M)$, $*$ approximated sv's $\hat{\sigma}_1^{(m-n)}, \ldots, \hat{\sigma}_n^{(m-n)}$,$\circ$ dismissed sv's $\mu_{n+1}, \ldots, \mu_m$

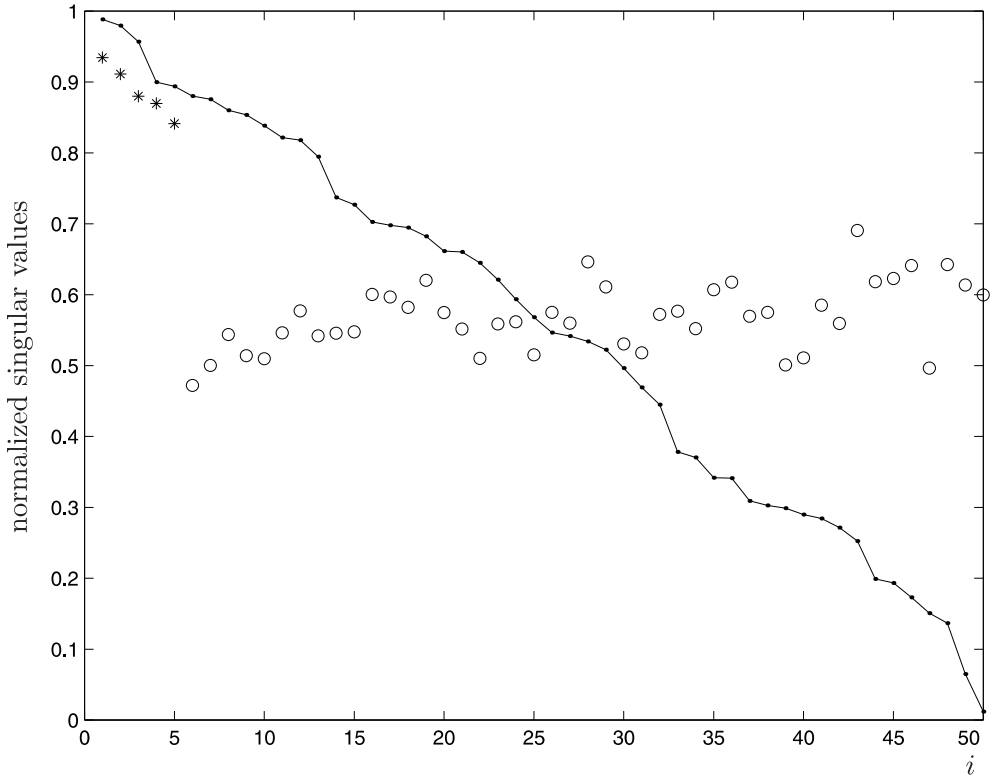


Fig. 4.1. The effect of the gap on the quality of the approximation ($\gamma = 0.01375$)

σ_i	$\hat{\sigma}_i^{(m-n)}$	$\lvert\sigma_i - \hat{\sigma}_i\rvert$	$\dfrac{\hat{\mu}^2}{(2\hat{\sigma}_i^{(m-n)})}$	$\cos\theta_i$	$\cos\hat{\theta}_i$	$\cos\phi_i$	$\cos\hat{\phi}_i$
0.98833	0.93436	0.05398	0.27320	0.97419	0.36164	0.95272	0.34189
0.97975	0.91122	0.06852	0.28725	0.94833	0.11482	0.91511	0.10679
0.95684	0.87986	0.07698	0.30809	0.88082	0.04148	0.84415	0.03815
0.89977	0.86969	0.03008	0.31534	0.80644	0.11320	0.75753	0.10941
0.89390	0.84136	0.05253	0.33693	0.16487	0.27966	0.14274	0.26322

$\mu = 0.97905$	$\hat{\mu} = 0.69067$	$\sigma_{n+1} = 0.88014$

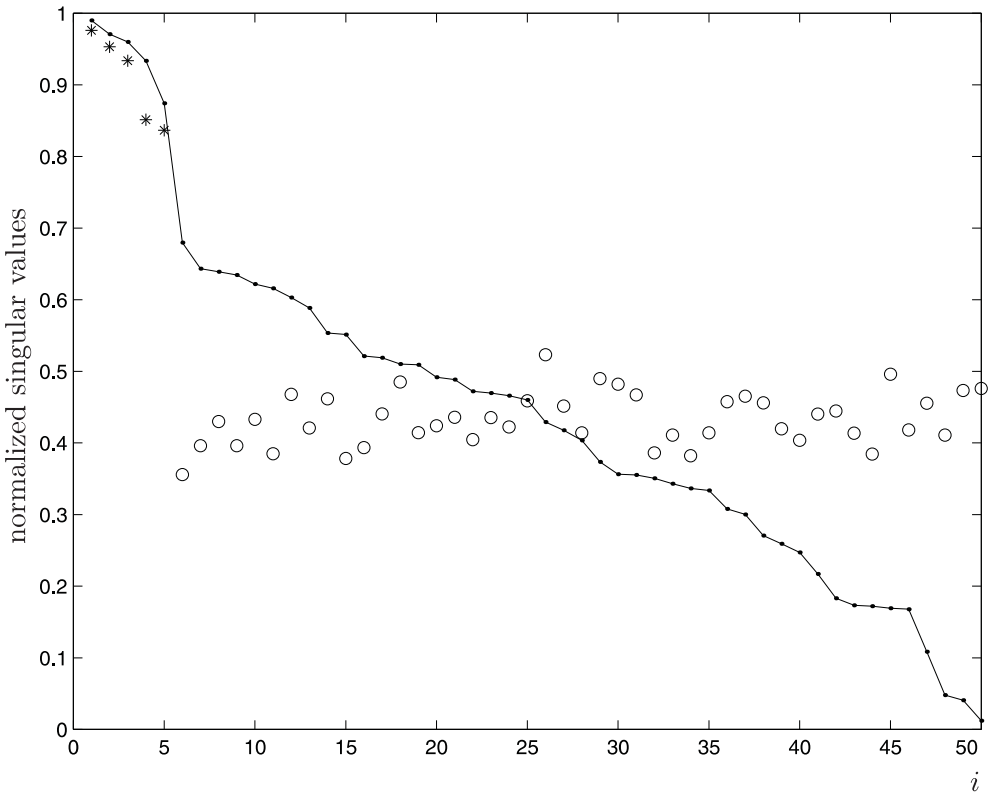


Fig. 4.2. The effect of the gap on the quality of the approximation ($\gamma = 0.19458$)

σ_i	$\hat{\sigma}_i^{(m-n)}$	$\lvert\sigma_i - \hat{\sigma}_i\rvert$	$\dfrac{\hat{\mu}^2}{(2\hat{\sigma}_i^{(m-n)})}$	$\cos\theta_i$	$\cos\hat{\theta}_i$	$\cos\phi_i$	$\cos\hat{\phi}_i$
0.99008	0.97613	0.01395	0.14370	0.99778	0.94233	0.99114	0.92905
0.97084	0.95301	0.01783	0.15075	0.99674	0.50515	0.98802	0.49587
0.96010	0.93379	0.02631	0.15703	0.98935	0.49785	0.95970	0.48421
0.93338	0.85142	0.08196	0.18888	0.97361	0.74349	0.92863	0.67820
0.87437	0.83675	0.03762	0.19556	0.93001	0.78790	0.83881	0.75401

$\mu = 0.73768$	$\hat{\mu} = 0.52330$	$\sigma_{n+1} = 0.67978$

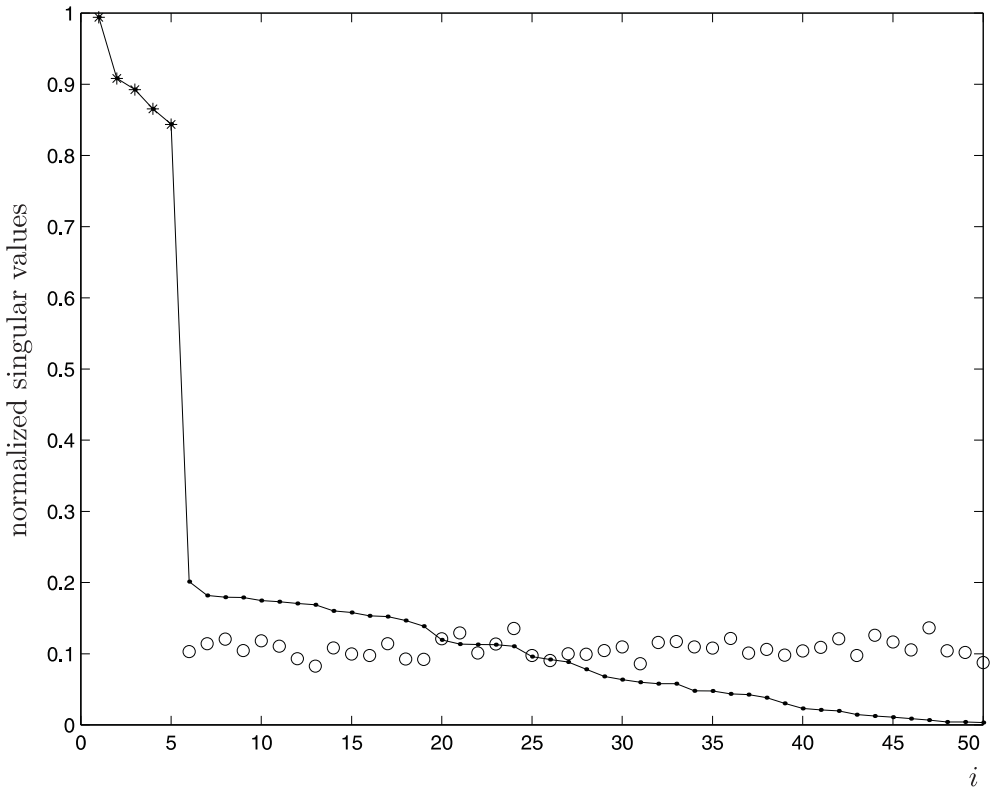


Fig. 4.3. The effect of the gap on the quality of the approximation ($\gamma = 0.64265$)

σ_i	$\hat{\sigma}_i^{(m-n)}$	$\lvert\sigma_i - \hat{\sigma}_i\rvert$	$\dfrac{\hat{\mu}^2}{(2\hat{\sigma}_i^{(m-n)})}$	$\cos\theta_i$	$\cos\hat{\theta}_i$	$\cos\phi_i$	$\cos\hat{\phi}_i$
0.99430	0.99418	0.00011	0.00940	0.99999	0.99999	0.99996	0.99988
0.90840	0.90815	0.00024	0.01126	0.99999	0.99989	0.99995	0.99962
0.89284	0.89250	0.00034	0.01166	0.99999	0.99989	0.99983	0.99950
0.86560	0.86551	0.00009	0.01240	0.99999	0.99998	0.99964	0.99987
0.84387	0.84357	0.00030	0.01305	0.99998	0.99998	0.99935	0.99963

$\mu = 0.20140$	$\hat{\mu} = 0.13631$	$\sigma_{n+1} = 0.20121$

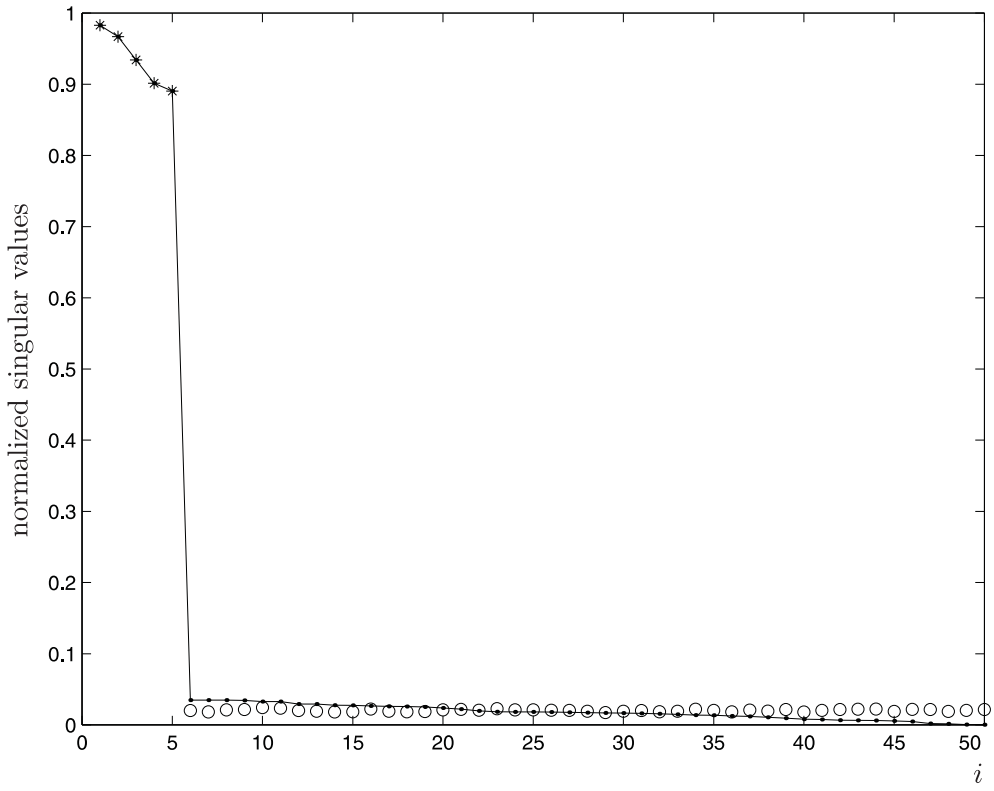


Fig. 4.4. The effect of the gap on the quality of the approximation ($\gamma = 0.85541$)

σ_i	$\hat{\sigma}_i^{(m-n)}$	$\lvert\sigma_i - \hat{\sigma}_i\rvert$	$\frac{\hat{\mu}^2}{(2\hat{\sigma}_i^{(m-n)})}$	$\cos\theta_i$	$\cos\hat{\theta}_i$	$\cos\phi_i$	$\cos\hat{\phi}_i$
0.98299	0.98299	$2.0\ 10^{-7}$	0.00030	0.99999	0.99999	0.99999	0.99999
0.96689	0.96689	$1.0\ 10^{-7}$	0.00032	0.99999	0.99999	0.99999	0.99999
0.93424	0.93424	$1.0\ 10^{-7}$	0.00034	0.99999	0.99999	0.99999	0.99999
0.90161	0.90161	$0.5\ 10^{-7}$	0.00036	0.99999	0.99999	0.99999	0.99999
0.89032	0.89032	$1.5\ 10^{-7}$	0.00037	0.99999	0.99999	0.99999	0.99999

$\mu = 0.03491$	$\hat{\mu} = 0.02430$	$\sigma_{n+1} = 0.03491$

4.6 The effect of round-off

In this section we analyze the propagation of round-off in the proposed algorithm for the worth case $k = 1$ (i.e., we add just one column at each iteration). For the block case, one can do the analysis successively for each column. The first aim is to prove some kind of backward stability of the algorithm. We show that at each step i the algorithm produces "approximate" matrices $\bar{V}_{(i)}, \bar{Q}_{(i)}$, and $\bar{R}_{(i)}$ that satisfy exactly the perturbed equations

$$[M(:, 1 : n + i) + E]\bar{V}_{(i)} = \bar{Q}_{(i)}\bar{R}_{(i)}, \quad (\bar{V}_{(i)} + F)^*(\bar{V}_{(i)} + F) = I_n, \tag{4.36}$$

where

$$\|E\|_F \leq \epsilon_e \|M\|_2, \ \ \epsilon_e \approx \epsilon_m, \quad \|F\|_F \leq \epsilon_f \approx \epsilon_m,$$

in which ϵ_m is the so-called unit round-off of the IEEE floating point standard (§.1.1.5). This is used to prove that the effect of round-off remains small despite the fact that this is a classical Gram-Schmidt procedure. The proof of the following theorem is given in the Appendix.

Theorem 4.5.

The recursive algorithm described in Sections 4.1 and 4.2 produces "approximate" matrices $\bar{V}_{(i)}$, $\bar{Q}_{(i)}$, and $\bar{R}_{(i)}$ that satisfy exactly *the perturbed equation (4.36) with the bounds (up to $O(\epsilon_m^2)$ terms) :*

$$\|E\|_F \leq \epsilon_e \|M\|_2, \;\; \epsilon_e \leq 26 n^{\frac{3}{2}} m \epsilon_m, \quad \|F\|_F \leq \epsilon_f \leq 9 n^{\frac{3}{2}} m \epsilon_m.$$

■

We point out here that these bounds do *not* depend on N, the largest dimension of M. Moreover, if one uses Householder transformations rather than Givens transformations, the results are very similar.

Remark 4.6.

Although Theorem 4.5 indicates that the error $\|E\|_F$ grows with the number of columns m, it does not seem to grow in actual experiments. This can be explained as follows. Assume that at step i we have the perturbed equation

$$\left[Q_{(i-1)} + E_{(i-1)} \; \hat{q}_i + e_i \right] G_u = \left[Q_{(i)} + E_{(i)} \; q_i + g_i \right], \tag{4.37}$$

where $E_{(i)}$ accounts for the loss of orthogonality in $Q_{(i)}$, and e_i is the local error in the vector $\hat{q}_i$, and g_i is the resulting error in the vector q_i. If we assume the errors in the right-hand side of (4.37) to be evenly distributed over the matrix, then it follows that

$$\|E_{(i)}\|_F^2 \leq \frac{n}{(n+1)} \|E_{(i-1)}\|_F^2 + \|e_i\|_2^2, \tag{4.38}$$

which for growing i tends to a limit

$$\|E\|_F^2 \leq (n+1) \max_i \|e_i\|_2^2$$

that is independent of m. The same reasoning can be applied to the error $\|F\|_F$. The corresponding bounds of Theorem 4.5 become

$$\epsilon_e \leq 26 n^2 \epsilon_m, \quad \epsilon_f \leq 9 n^2 \epsilon_m.$$

We now turn our attention to the loss of orthogonality in the computed matrix $\bar{Q}$. This can be bounded using a perturbation result for the QR factorization of

$$(M+E)\bar{V} = M\bar{V} + E\bar{V} \doteq M\bar{V} + G,$$

where, using the bounds of Theorem 4.5, we have :

$$\|G\|_F = \epsilon_g \|M\|_2, \quad \epsilon_g \leq \epsilon_e + O(\epsilon_e \epsilon_f) \approx \epsilon_m.$$

Theorem 4.7.

Let (a given matrix) $\bar{V} \in \mathcal{R}^{N \times n}$ "select" n columns of the matrix $M \in \mathcal{R}^{N \times m}$, and let

$$M\bar{V} = QR, \quad Q^*Q = I_n,$$

with R upper triangular, be its exact QR factorization. Let

$$M\bar{V} + G = \bar{Q}\bar{R}, \quad \|G\|_F = \epsilon_g \|M\|_2 \approx u \|M\|_2 \tag{4.39}$$

be a "computed" version, where $\bar{Q} = Q + \Delta_Q$, $\bar{R} = R + \Delta_R$. Then under a mild assumption, namely, condition (4.41), we can bound the loss of orthogonality in $\bar{Q}$ as follows :

$$\|\bar{Q}^*\bar{Q} - I_n\|_F \leq \sqrt{2}\epsilon_g \kappa_2(R)\kappa_R(M\bar{V}) \leq 2\epsilon_g \kappa_2^2(R), \quad \epsilon_g \approx \epsilon_m.$$

■

Proof. Since $\bar{Q}$ is not necessarily orthogonal we first compute its QR factorization :

$$\bar{Q} = Q_0 R_0, \quad Q_0^* Q_0 = I_n.$$

So we can consider the perturbation of the QR decomposition of $M\bar{V}$:

$$M\bar{V} = QR, \quad M\bar{V} + G = Q_0(R_0\bar{R}). \tag{4.40}$$

The loss of orthogonality in $\bar{Q}$ can be measured by R_0 since

$$\bar{Q}^*\bar{Q} - I_n = R_0^* Q_0^* Q_0 R_0 - I_n = R_0^* R_0 - I_n.$$

To measure this, we first use a perturbation analysis of [41] for (4.40) to obtain

$$\|R_0\bar{R} - R\|_F \leq \epsilon_g \kappa_R(M\bar{V})\|R\|_2,$$

where $\kappa_R(M\bar{V})$ is the "refined" condition number of the factor R of the QR factorization (4.40) of $S_k^S\bar{V}$ [41]. If we define $\Delta_0 \doteq R_0 - I_n$, we then have

$$R_0\bar{R} - R = (I_n + \Delta_0)(R + \Delta_R) - R = \Delta_0\bar{R} + \Delta_R \approx \Delta_0 R + \Delta_R$$

and, hence,

$$\|\Delta_0 R + \Delta_R\|_F \approx \|\Delta_0\bar{R} + \Delta_R\|_F \leq \epsilon_g \kappa_2(R)\|R\|_2.$$

We now assume that there are no strong cancellations between $\|\Delta_R\|_F$ (measuring the perturbation of R) and $\|\Delta_0 R\|_F$ (measuring the perturbation in Q) and hence that $\|\Delta_0 R\|_F$ and $\|\Delta_R + \Delta_0 R\|_F$ are of the same order of magnitude :

$$\|\Delta_0 R\|_F \approx \|\Delta_R + \Delta_0 R\|_F. \tag{4.41}$$

From $\|\Delta_0 R\|_F \leq \epsilon_g \kappa_R(A\bar{V})\|R\|_2$ it then follows that

$$\|\Delta_0\|_F \leq \epsilon_g \kappa_R(A\bar{V})\|R\|_2\|R^{-1}\|_2.$$

This can now be used to bound

$$\|R_0^* R_0 - I_n\|_F = \|\Delta_0 + \Delta_0^* + \Delta_0^*\Delta_0\|_F \approx \sqrt{2}\|\Delta_0\|_F,$$

which yields

$$\|R_0^* R_0 - I_n\|_F \leq \sqrt{2}\epsilon_g \kappa_2(R)\kappa_R(S_k^S\bar{V}). \tag{4.42}$$

Using the overestimate $\kappa_R(S_k^S\bar{V}) \leq \sqrt{2}\kappa_2(R)$ of [41] we approximate this finally by

$$\|R_0^* R_0 - I_n\|_F \leq 2\epsilon_g \kappa_2^2(R). \tag{4.43}$$

□

Remark 4.8.

Assumption (4.41) is crucial to the proof of Theorem 4.7. It is easy to see that any factorization of the type (4.39) will not yield the bounds (4.42) or (4.43) : consider, e.g., the factorization

$$M\bar{V} + G = (\bar{Q}U)(U^{-1}\bar{R}),$$

where U is any invertible upper triangular matrix. This clearly satisfies the conditions of the theorem, except for assumption (4.41). The critical quantity for this new factorization then becomes $\|U^* R_0^* R_0 U - I_n\|_F$, and since U can be chosen arbitrarily, it is impossible to bound it. Assumption (4.41) is therefore crucial, and we show in the next section that it indeed holds in practice.

4.6.1 Numerical tests for the error propagation

In this section we present numerical evidence that the analysis of the previous section can be applied to the tracking problem of the dominant spaces of a given matrix. The numerical experiments we ran show that the loss of orthogonality in the computed matrix $\bar{Q}_{(i)}$ of (4.36) remains bounded by the squared condition number of the matrix R that we are "tracking".

We show in Figures 4.5 - 4.8 four plots that compare the loss of orthogonality in the proposed algorithm based on the classical Gram-Schmidt method (labeled CGS) and a "fully orthogonal" method, which we obtain by performing *two* steps of CGS, rather than one, at each iteration. This second method, labeled CGS2, was analyzed in [2] and shown to yield a Q factor that is close to orthogonal. We chose this as an alternative to the Householder method because in the iterative scheme considered here, CGS2 involves significantly fewer operations than the Householder method.

As suggested by Remark 4.6, the backward error $E_{(i)}$ and the quantity ϵ_e can be bounded independently of the step i. We therefore compare the loss of orthogonality $\|R_0^* R_0 - I_n\|_F$ with the quantities $\epsilon_m n^2 \kappa_2(R_{(i)}) \kappa_R(S^S_{i+\zeta-1} \bar{V}_{(i)})$ for the CGS method and $\epsilon_m n^2$ for the CGS2 method. These "simplified" quantities are indicators to show that the loss of orthogonality is of the order of magnitude predicted by our error analysis. To show the effect of the condition number of the triangular factor $R_{(i)}$, we let it grow in the four examples by choosing a growing condition number for S^S_k.

Legend for Figures 4.5 - 4.8 :
_ _ CGS bound $\epsilon_m n^2 \kappa_2(R_{(i)}) \kappa_R(A(:, 1:i)\bar{V}_{(i)})$, ___ CGS2 bound $\epsilon_m n^2$,
$*$ loss of orthogonality in CGS method, $\circ$ loss of orthogonality in CGS2 method

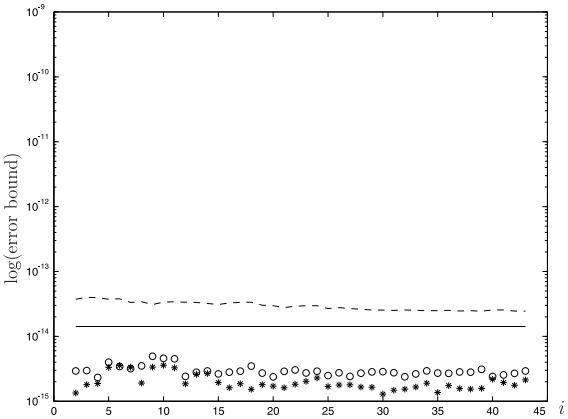


Fig. 4.5. Error propagation for the case
$\gamma = 0.01375$
$\kappa_2(A) = 41.806$, $\kappa_2(R_{(\tau)}) = 1.156$,
$\kappa_R(A\bar{V}_{(\tau)}) = 1.492$.

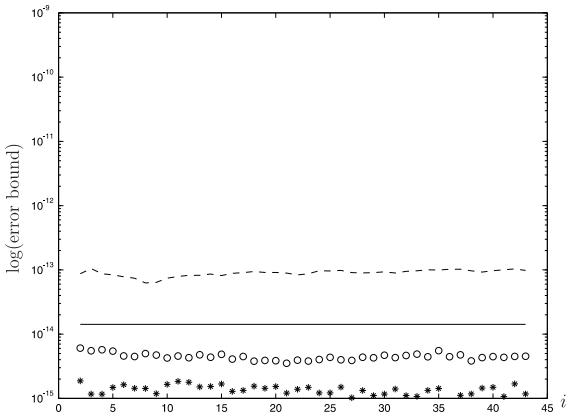


Fig. 4.6. Error propagation for the case
$\gamma = 0.19458$.
$\kappa_2(A) = 165.4$, $\kappa_2(R_{(\tau)}) = 2.643$,
$\kappa_R(A\bar{V}_{(\tau)}) = 2.613$.

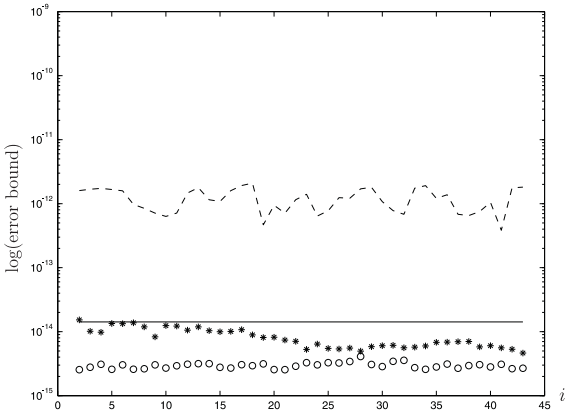


Fig. 4.7. Error propagation for the case
$\gamma = 0.64265$.
$\kappa_2(A) = 214.3$, $\kappa_2(R_{(\tau)}) = 12.04$,
$\kappa_R(A\bar{V}_{(\tau)}) = 10.60$.

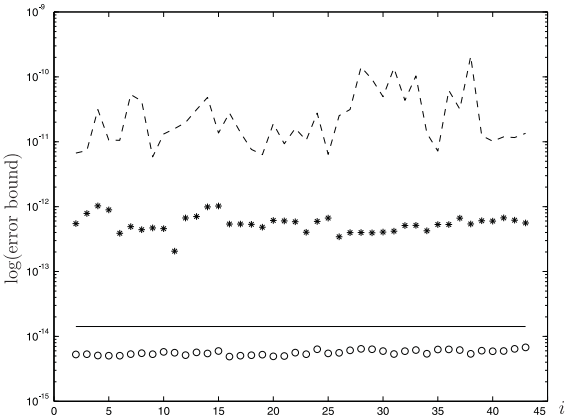


Fig. 4.8. Error propagation for the case
$\gamma = 0.85541$.
$\kappa_2(A) = 6928$, $\kappa_2(R_{(\tau)}) = 134.7$,
$\kappa_R(A\bar{V}_{(\tau)}) = 7.028$.

The following observations can be derived from these experiments :

- The condition numbers $\kappa_2(R_{(i)})$ and $\kappa_R(A(:, 1\colon i)\bar{V}_{(i)})$ do not affect the loss of orthogonality of the CGS2 method, as expected from the analysis of [2]. The product $\kappa_2(R_{(i)})\kappa_R(A(:, 1 : i)\bar{V}_{(i)})$ can be inferred from the gap between the CGS and CGS2 bounds.
- The statistical assumption of Remark 4.6 seems to hold since there is no growth in the loss of orthogonality of the computed matrices $\bar{Q}_{(i)}$: this should depend on the backward error $E_{(i)}$, which does not depend on i if the assumption of Remark 4.6 holds
- Assumption (4.41) made in Theorem 4.7 was verified in these experiments and validates the resulting bounds (4.42), (4.43) of that theorem; the graphs in Figure 4.9 give the norms of the two quantities for the two examples given earlier and illustrate that the assumption that those quantities are of the same order of magnitude is reasonable.

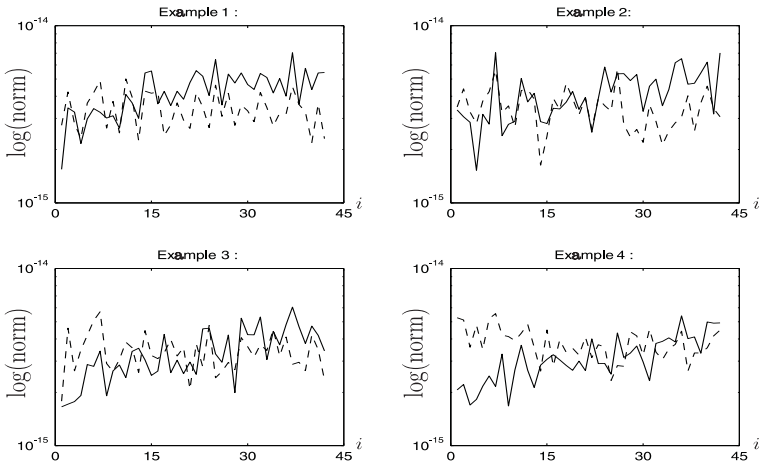


Fig. 4.9. Verification of assumption (4.41) for examples Figures 4.5–4.8.
– – $\|\Delta_0 R\|_F$, ——— $\|\Delta_R + \Delta_0 R\|_F$.

- The loss of orthogonality remains very reasonable when the condition number $\kappa_2(R_{(i)})$ is not too large, which is a reasonable assumption in applications where a "dominant matrix" $R_{(i)}$ is being tracked.

We observed no difference in the computed spaces for the CGS or CGS2 methods. We conclude from our analysis and the experimental evidence that the cheapest version of the algorithm (CGS) can be used safely for the Modified Low-Rank Smith (LMRS) algorithm. By this we mean that the angles $\cos\theta_k$ and $\cos\phi_k$ for both methods were equal in the first four digits despite a very small loss of orthogonality in the CGS method.

4.7 Convergence properties of the Modified Low-Rank Smith algorithm

The idea of our Modified Low-Rank Smith algorithm for model reduction is to apply the previous procedure (Algorithm 5) to the controllability and observability matrices

$$\mathcal{C}_i = \left[\, B \; AB \; A^2B \; \ldots \; A^{i-1}B \,\right],$$
$$\mathcal{O}_i^T = \left[\, C^T \; A^TC^T \; (A^2)^TC^T \; \ldots \; (A^{i-1})^TC^T \,\right],$$

to obtain their respective best low-rank approximations S_i^{MLRS} and R_i^{MLRS}. These low-rank approximations are used to provide approximations of the Gramians.
It is obvious that the quality of the approximation of the controllability and observability matrices, and so the Gramians themselves will influence the quality of our reduced model obtained via these approximations. In this section we compare the exact solutions with their approximations. We focus on the controllability matrix and Gramian; the same results hold for the observability elements.

Let us consider the decomposition (4.15) :

$$\mathcal{C}_iV_{\tau_i} = \left[\, Q_{(\tau_i)}R_{(\tau_i)} \; Q_{(\tau_i)_{res}}R_{(\tau_i)_{res}} \; \ldots \; Q_{(1)_{res}}R_{(1)_{res}} \,\right].$$

We can use this to compare the Gramian $\mathcal{G}_c(i) = \mathcal{C}_i\mathcal{C}_i^T$ with its approximation $\mathcal{P}_i = S_i^{MLRS}(S_i^{MLRS})^T$ as follows

$$\mathcal{C}_i\mathcal{C}_i^T = S_i^{MLRS}(S_i^{MLRS})^T + \sum_{j=1}^{i-1} Q_{(\tau_j)_{res}}R_{(\tau_j)_{res}}R^*_{(\tau_j)_{res}}Q^*_{(\tau_j)_{res}}.$$

Taking norms we obtain

$$\begin{aligned}
\|\mathcal{C}_i\mathcal{C}_i^T - S_i^{MLRS}(S_i^{MLRS})^T\|_2 &\leq \|\sum_{j=1}^{i-1} Q_{(\tau_j)_{res}}R_{(\tau_j)_{res}}R^*_{(\tau_j)_{res}}Q^*_{(\tau_j)_{res}}\|_2 \\
&\leq \sum_{j=1}^{i-1} \|R_{(\tau_j)_{res}}R^*_{(\tau_j)_{res}}\|_2 \\
&\leq \sum_{j=1}^{i-1} (\mu_1^{(j)})^2.
\end{aligned}$$

If we define

$$\eta_c = \max_{1\leq j\leq \tau} \mu_1^{(j)},$$

where τ is the number of iterations, we obtain the simple inequality

$$\|\mathcal{C}_i\mathcal{C}_i^T - S_i^{MLRS}(S_i^{MLRS})^T\|_2 \leq (i-1)\eta_c^2. \tag{4.44}$$

This bounds the difference between $\mathcal{C}_i\mathcal{C}_i^T$ and $S_i^{MLRS}(S_i^{MLRS})^T$ for all i in terms of the "noise" level η_c. Our bound is very similar to those obtained in [67] (see §.3.2).
In this paper, a similar study is made, but there is a difference between our results and those obtained in [67]. First, their bounds are a priori bounds instead of our a posteriori bound which depends mainly of the "noise" level $\mu_\bullet$. Moreover, we do not compute it exactly, but just estimate it with $\hat{\mu}_\bullet$. The quality of our approximation depends dramatically of this "noise" level, the gap between the dismissed singular values and the kept ones, and finally of the stability of the matrix A of the system.

4.8 Application and numerical examples

Previous sections and previous research (e.g. [67] for model reduction, and [40] for image analysis) have already demonstrated that the eigenspace approach is a powerful tool. Our objective here is to illustrate the efficiency of our incremental update algorithm. This is done with two sets of numerical examples : a first one in image analysis and a second one in model reduction. These two examples show that the quality of the resulting approximations depends crucially of the gap between the kept part and the neglected one.

Remark 4.9. The recursive algorithms have been implemented using SLICOT functions linked to MATLAB6.1. We used an AMD Athlon processor which is running at 2GHz. It has a data cache of 512Ko, and a SDRam of 512Mo (certified 333Mhz PC2700).

4.8.1 Image analysis

The first set of experiments compares the reconstruction and approximation of the proposed incremental update algorithm, denoted ASVD (for *Adaptive Singular Value Decomposition*) with that of the optimal algorithm on sequences of images. The optimal algorithm performs a single SVD on the matrix containing all images in the ensemble. Hence, it represents the best case scenario in terms of approximation and reconstruction performance and serves as the baseline for comparison. Two particular sets, namely "Mother" and "Table" (we show some extracted images from the Mother sequence in Figure 4.10 and from the Table sequence in Figure 4.11), are used here.
The first represents a mother and her daughter. The mother is the only person who moves in the sequence. The second sequence shows two ping-pong players. This sequence has an interesting characteristic which represents a certain difficulty for the approximation : the movement is very fast as opposed to the first sequence where the movement is very slow and localized. Both sequences contain 300 images. Each image consists of 288×352 pixels, and each pixel has 256 gray levels. We cut up each image into 352 columns and then we form one vector of dimension 101376 for each image. The matrix M_S has thus dimensions 101376×300. Figure 4.12 shows this in more detail.

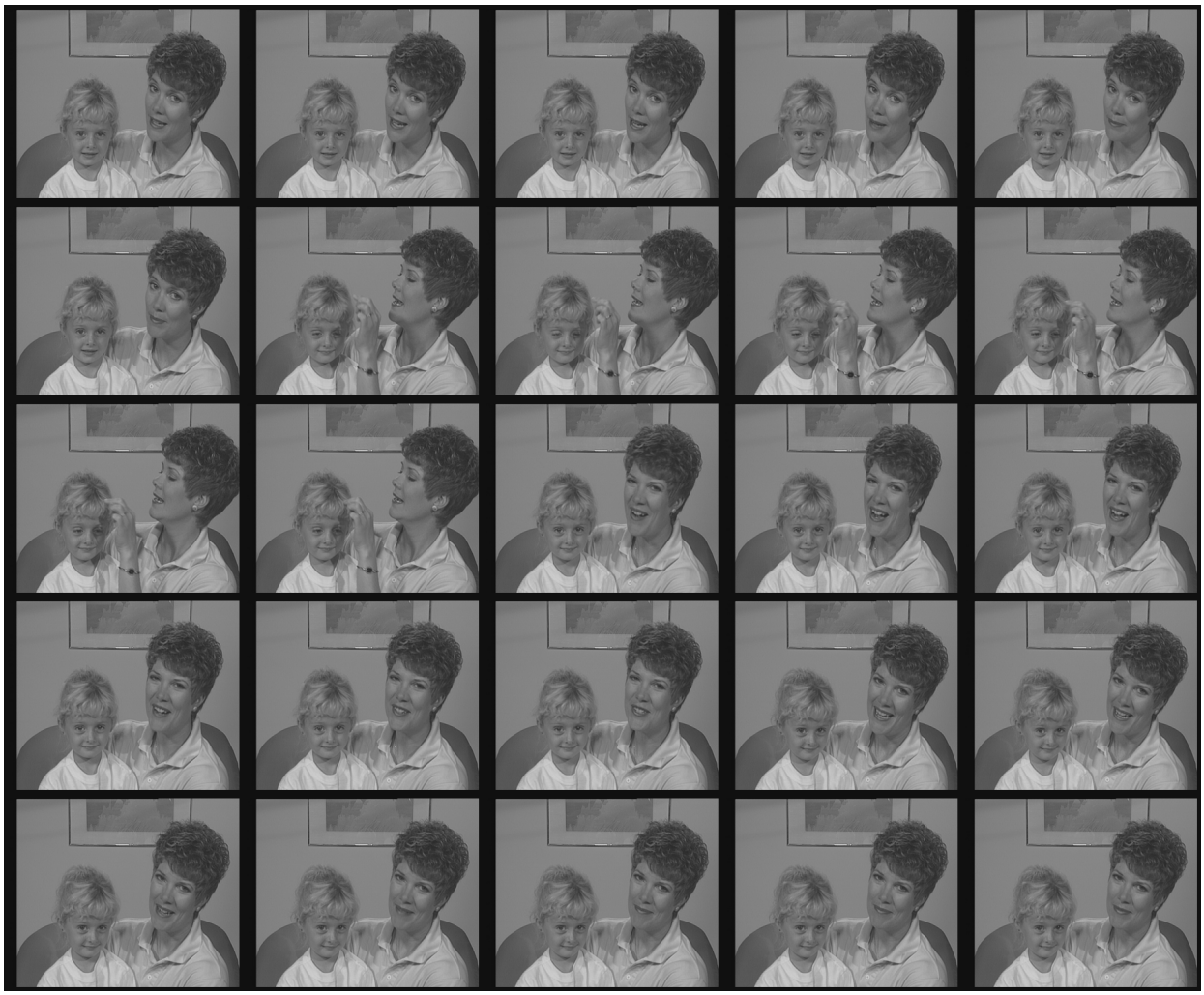


Fig. 4.10. Some images extracted from the Mother sequence.

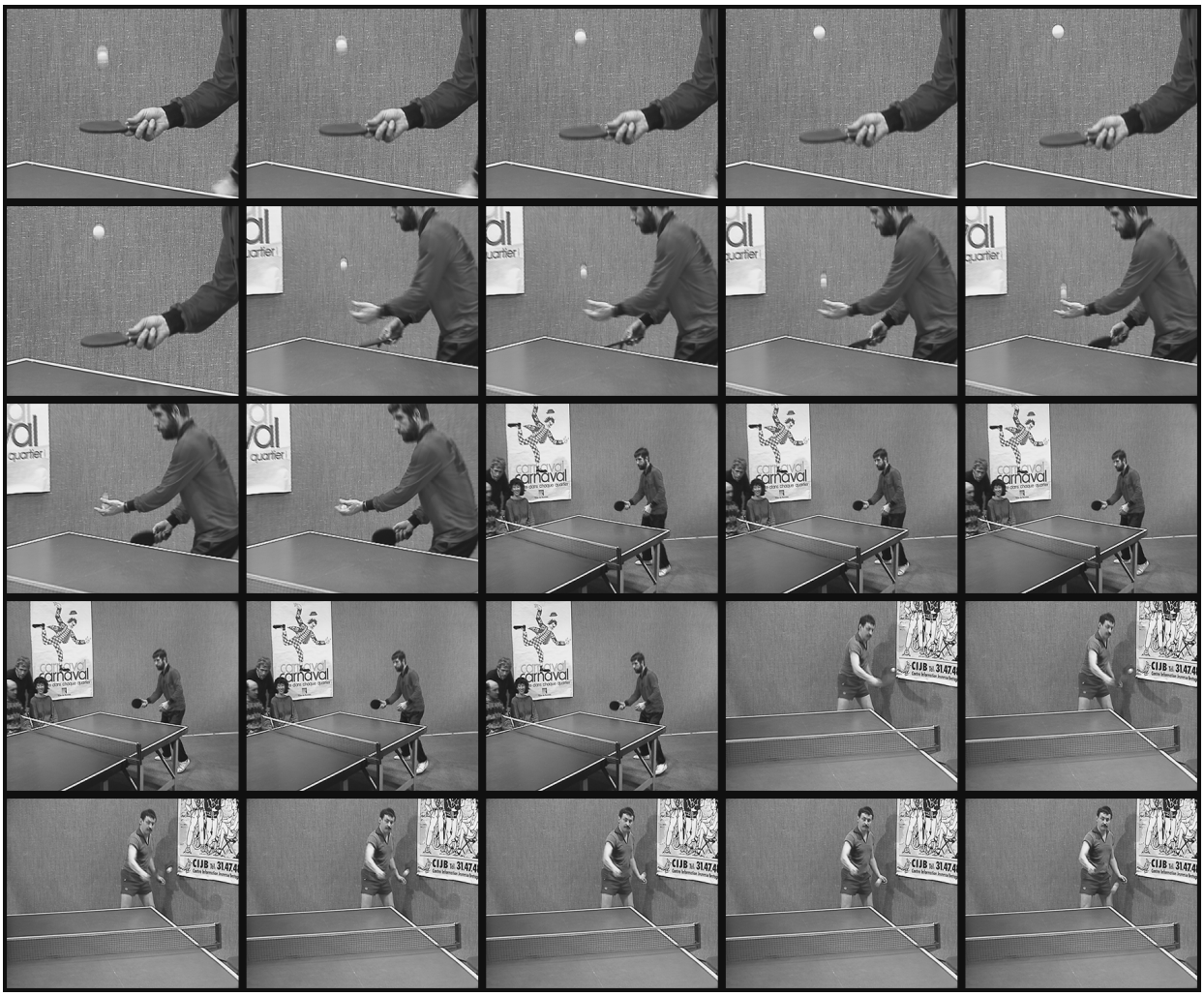


Fig. 4.11. Some images extracted from the Table sequence.

Handling directly the whole matrix M_S is very costly in time and memory. The run time of its SVD is about 1200 seconds while the construction of the matrix takes about 350 seconds. The total run time is thus about 1550 seconds and uses at least 245 Megabytes. Furthermore, the batch algorithm is not suitable for application in a dynamic environment because the inclusion of a single new image into the image set can require a complete recomputation of the basis set. The proposed updating (ASVD) algorithm easily handles any number of new images in an incremental manner, without recomputing the basis set from scratch, which has applications in active vision [40].

In the following experiments, comparisons are made using basis sets whose dimensions are $n = 10$, 20, 30, and 60 which correspond respectively to 3.3%, 6.6%, 10%, and 20% of the number of images in the sequence. In Table 4.2 we show the cost in time and memory used by the incremental updating algorithm for different values of n

	$n = 10$	$n = 20$	$n = 30$	$n = 60$
run time (in sec.)	225	393	548	960
memory used	8M	16M	25M	50M

Table 4.2. Run time and memory used for ASVD for different values of n.

At the top of Figures 4.13 and 4.14 we show an original image. We show in the two columns the reconstructed images with different percentage basis using the batch update (column left) and the adaptive SVD (column right). The examples show that the visual reconstruction quality for the incremental update is comparable to that of the batch algorithm. From our experiments, a 10% basis set is often sufficient to reconstruct images with an acceptable visual quality. For the Table example, the small table tennis ball is not quite distinguishable for the two algorithms we have just a ghost trace. This phenomenon is due to the fact that we collect the dominant behaviour of the sequence and so we have a certain history of the sequence which influences the reconstruction. For the Mother sequence this is not visible because the movement is slow and localized.
These examples show that the performance of the incremental SVD algorithm closely mimics that of the batch algorithm, but the difference lies in the cost as shown in Table 4.2.

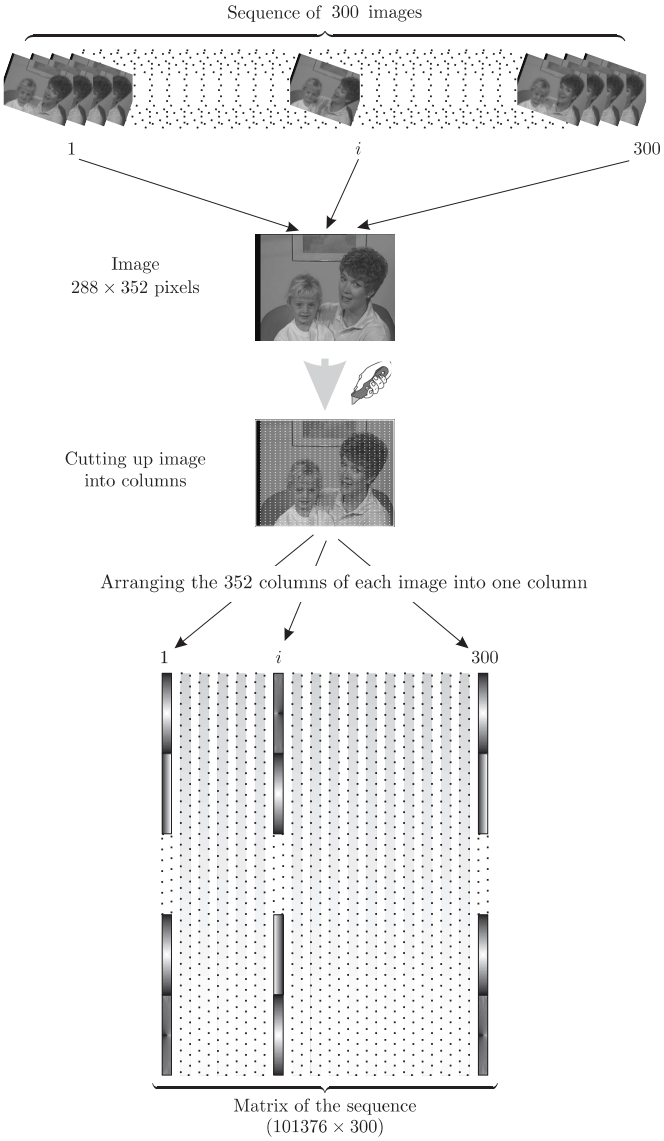


Fig. 4.12. Transforming the sequence of images into a matrix.

Original image.

Approx. using SVD(10).

Approx. using ASVD(10).

Approx. using SVD(20).

Approx. using ASVD(20).

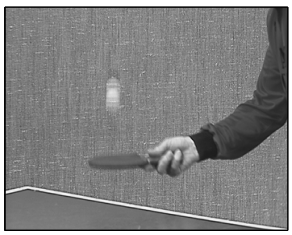
Approx. using SVD(30).

Approx. using ASVD(30).

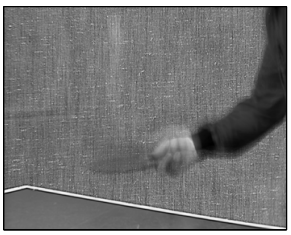
Approx. using SVD(60).

Approx. using ASVD(60).

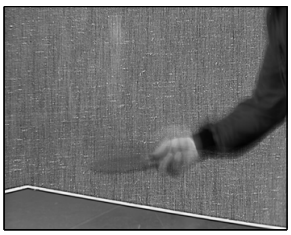
Fig. 4.13. Approximation of image using SVD and ASVD for different values of n.



Original image.



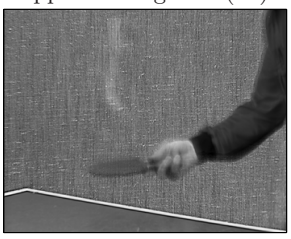
Approx. using SVD(10).



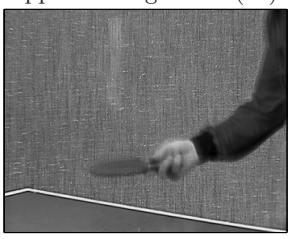
Approx. using ASVD(10).

Approx. using SVD(20).

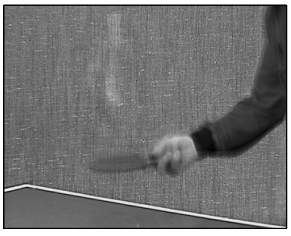
Approx. using ASVD(20).



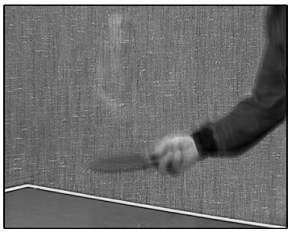
Approx. using SVD(30).



Approx. using ASVD(30).



Approx. using SVD(60).



Approx. using ASVD(60).

Fig. 4.14. Approximation of image using SVD and ASVD for different values of n.

4.8.2 Model reduction

In this section we apply our algorithm to four different dynamical systems : a Building model, a CD Player model, and two International Space Station model. These benchmarks are described in more details in [33] and [66]. These models are continuous so we use discretization with $\zeta = 2$ (see §.1.2.9).

Building model

This is the model of a building (the Los Angeles University Hospital) with 8 floors each having 3 degrees of freedom, namely displacements in x and y directions, and rotation.
We have then a second order system of 24 variables (see Chapter 7 for this kind of systems), and a state-space model of order 48, we are mostly interested in the motion in the first coordinate, and the output is the variation of this coordinate.
The reduced order is 10 corresponding to a relative tolerance value[2] equal to 0.16.

CD Player

The full order model describes the dynamics between the lens actuator and the radial arm position of a portable CD player. The model has 120 states with 2 inputs and 2 outputs. The reduced order is 24 corresponding to a relative tolerance value equal to $2.8.10^{-7}$.

International Space Station (ISS)

The International Space Station (ISS) is an important challenge for the control community. Its assembly and operation poses many problems due to its complex, variable flexible structure as well as a variety of operational modes and control systems. Hence, in order to certify the vehicle for flight readiness, it is of critical importance that all possible simulations and checks on robust stability and performance assessment are done.
Here, we consider the two first assembly stages 1R and 12A.

Stage 1R : This is a model of stage 1R (Russian Service Module). It has 270 states, 3 inputs and 3 outputs. We approximate the system with a model of order 32 corresponding to a relative tolerance value of 2.10^{-3}.

Stage 12A : It has 1412 states, 3 inputs and 3 outputs. The reduced order is 195 corresponding to a relative tolerance value of 65.10^{-4}.

The Hankel singular values of model 12A decay slower than those of model 1R (Figure 4.16).

Remark 4.10.

- Our recursive procedure (Algorithm 5) approximates the square roots of the Gramians by low-rank approximations. So we propose to use these low-rank approximations instead of the original square roots to provide an Approximated Balanced Truncation (see §.2.4).
- Actually, the three algorithms presented in this thesis were applied on all benchmark models in [33]. The largest model on which we applied our algorithms is the Modified Nodal Analysis model ($N = 10913$, $m = p = 9$). But the problem for very large models is that, even if our algorithms give results, we do not have any reference results to assess the performance of these algorithms. This is because MATLAB (even linked with SLICOT) and specially the Control Toolbox, which we use to compute the frequency response, can not handle such large systems. Actually, with this toolbox one can not handle a system of order more than 2000.
  The four benchmark models were especially chosen to illustrate the advantages and the drawbacks of the algorithms.

In Table 4.3 we give the order of the system (N), the number of inputs (m), and outputs (p), the order of reduced system (n), and the corresponding tolerance value. We show also in this table the spectral radius

[2] Relative tolerance value corresponding to a n^{th} order reduced system is given by the ratio $\frac{\sigma_n}{\sigma_1}$ where σ_1 and σ_n are the largest and n^{th} singular value of the system respectively.

and the condition number of the matrix A.
In Figures 4.15 and 4.16 we show the Hankel Singular Values (HSV) of our benchmark models.

	N	m	p	n	tol.value	$\rho(A)$	$\kappa(A)$
build model	48	1	1	10	0.16	0.4997	$8.0478.10^3$
CDplayer model	120	2	2	24	$2.8.10^{-7}$	0.5266	$1.7793.10^4$
ISS 1R model	270	3	3	32	2.10^{-3}	0.7338	$9.6802.10^3$
ISS 12A model	1412	3	3	195	65.10^{-4}	0.8310	$5.7728.10^3$

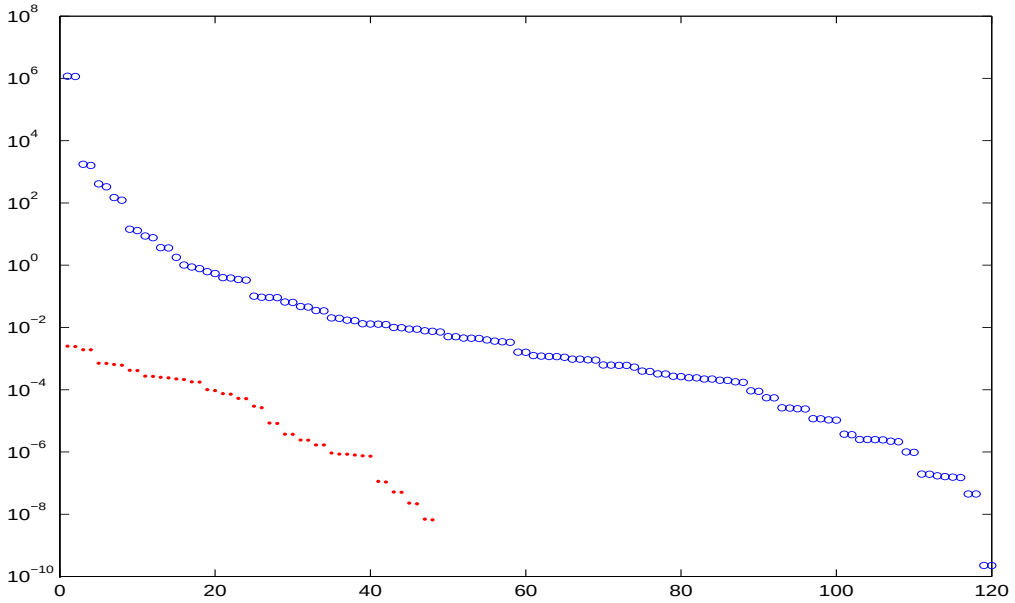


Fig. 4.15. Hankel Singular Values of benchmark models I.
• Building model, ○ CDplayer model.

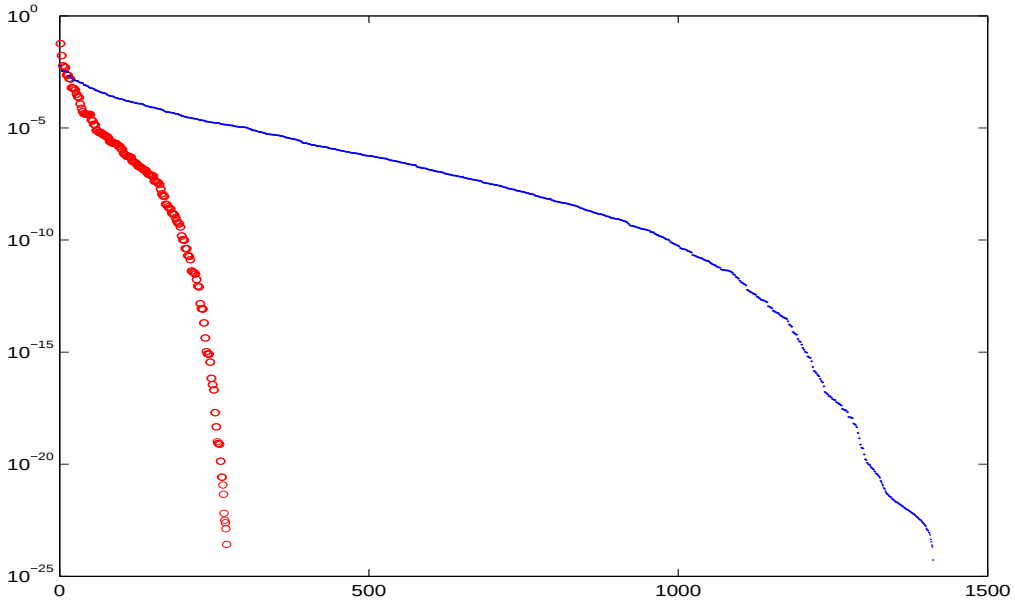


Fig. 4.16. Hankel Singular Values of benchmark models II.
○ ISS 1R model, • ISS 12A model.

For each example, the relative $\mathcal{H}_\infty$ norms of the error systems are given in Table 4.4; μ_c and μ_o are given in Table 4.5 and the CPU times for each algorithm are given in Table 4.6. The σ_{max}-plot of the full order and the corresponding error systems are shown in Figures 4.17 and 4.18 for the Building model, in Figures 4.19 and 4.20 for the CDplayer model, in Figures 4.21 and 4.22 for the ISS 1R model, and in Figures 4.23 and 4.24 for the ISS 12A model.

The σ_{max}-plot is obtained with computing the 2-norm (maximum singular value) of the frequency response of the considered system, evaluated at each frequency of the considered frequency meanwhile. The σ_{max}-plot of the considered error system gives the worst 2-norm of the frequency response of this error system. This is very useful especially for MIMO systems.

It can be seen from Figures 4.17, 4.19, 4.21, and 4.23 that we obtain results with the MLRS algorithm, which are not really close to those obtained using Balanced Truncation. This is due to the fact that there is no gap between the kept part and the neglected one (see Figures 4.15 and 4.16), unlike the image example where the approximation was quite better. In fact for the image example, we chose the order such there was a gap between the kept part and the neglected one. But here we chose only the tolerance value, and the neglected singular values can be close to the small value kept. This may explain the bad approximation. To avoid this, one can take a bigger value for n than the actual values which will improve the quality of the approximation.

We remark also that, in general, the obtained values for μ_c and μ_o are quite different and not of the same order, which means that the Gramians are not both well approximated at the same time. Actually, these noise levels are of the same order if the system is *nearly balanced*, i.e., the condition number of the balancing transformation T ($\kappa(T)$) is reasonable (relatively close to 1). For instance, only the CDplayer model is close balanced ($\kappa(T) = 40.7341$), and one can see that for this model we have $\mu_o \approx \mu_c$ (see Table 4.5).

The three other examples are poorly balanced. Actually, we have for the Building model ($\kappa(T) = 347.0781$), and for the International Space Station ($\kappa(T) = 7.4018.10^5$). It is shown in Table 4.5 that the resulting noise levels for each of these models are indeed not of the same order.

One can conclude that for poorly balanced systems the two Gramians are not well approximated at the same time, which yields a poorer quality of the reduced order model.

Also, even if $\mu_\bullet$ decrease rapidly (Figures 4.18, 4.20, 4.22, and 4.24), the quality of the approximations depends on the maximum value of $\mu_\bullet$ (see Table 4.5).

Another remark is that for the second model of International Space Station, the traditional Balanced Truncation function of MATLAB does not work (we receive a message of MATLAB "Out of memory" after one day of computation), so we use the SLICOT function which gives the result in a reasonable time. The reason is that the Balanced Truncation function of MATLAB does not take into account the Cholesky factor and sparsity of the system matrices. Moreover, this model is nearly uncontrollable and unobservable, which implies that when we have to compute the SVD of the controllability or observability matrices, we will not converge as we have a singularity. For the stability, in general, we obtain stable reduced-order models but close to instability, this is due mainly to the numerical errors and the condition number of the matrix A which is very large (see Table 4.3).

Indeed, the Modified Low-Rank Smith (MLRS) algorithm depends dramatically of the choice of the reduced order. This choice has to be done such that there is a gap between the kept part and the neglected one. Actually, this choice has to be done for both controllability and observability Gramians. This causes some problems if the gap in the singular values of one Gramian is not present in the singular values of the other Gramian at the same level, which means that one Gramian will not be well approximated and so we obtain a reduced order model of bad quality. This is the case for the poorly balanced systems.

model	$\|\mathcal{S}\|_\infty$	$\frac{\|\mathcal{S}-\mathcal{S}_n^{BT}\|_\infty}{\|\mathcal{S}\|_\infty}$	$\frac{\|\mathcal{S}-\mathcal{S}_n^{MLRS}\|_\infty}{\|\mathcal{S}\|_\infty}$
Building	0.0053	0.1143	0.9994
CDplayer	$2.3198.10^6$	$8.0704.10^{-8}$	0.9875
ISS 1R	0.1159	0.0013	1.0000
ISS 12A	0.0107	0.0071	0.9992

Table 4.4. H_∞ norm of benchmark models and the MLRS error systems.

model	μ_c^{MLRS}	μ_o^{MLRS}
Building	$3.4411.10^{-4}$	0.6655
CDplayer	6.3739	6.3234
ISS 1R	0.5494	0.0030
ISS 12A	0.8248	0.0029

Table 4.5. MLRS noise levels $\mu_\bullet$ for benchmark models.

model	BT	MLRS
Building	0.3750	0.1720
CDplayer	0.7970	1.8750
ISS 1R	11.6720	9.4530
ISS 12A	$1.1327.10^3$	$0.6794.10^3$

Table 4.6. CPU time for BT and MLRS.

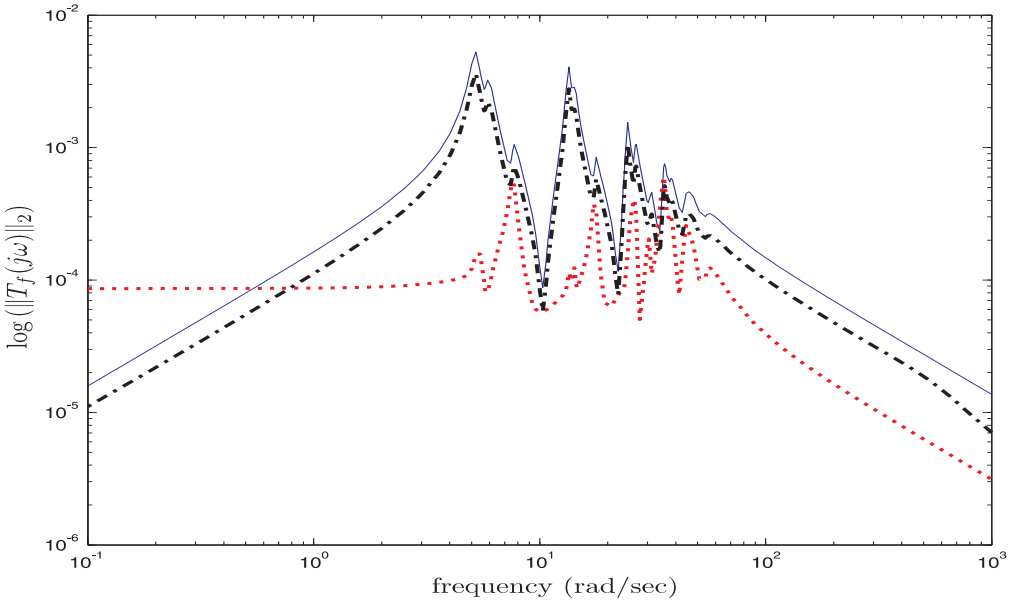


Fig. 4.17. $\sigma_{\max}$-plot of the frequency responses for Building model.
— full model, ··· BT error system, -·- MLRS error system.

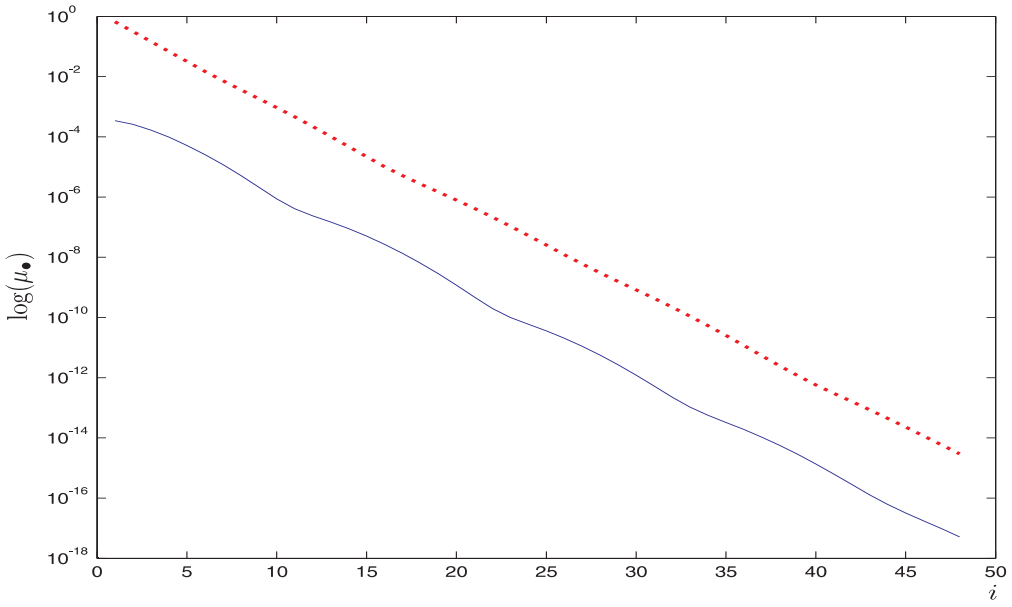


Fig. 4.18. Evolution of the values of $\mu_\bullet$ for Building model.
— μ_c and ··· μ_o.

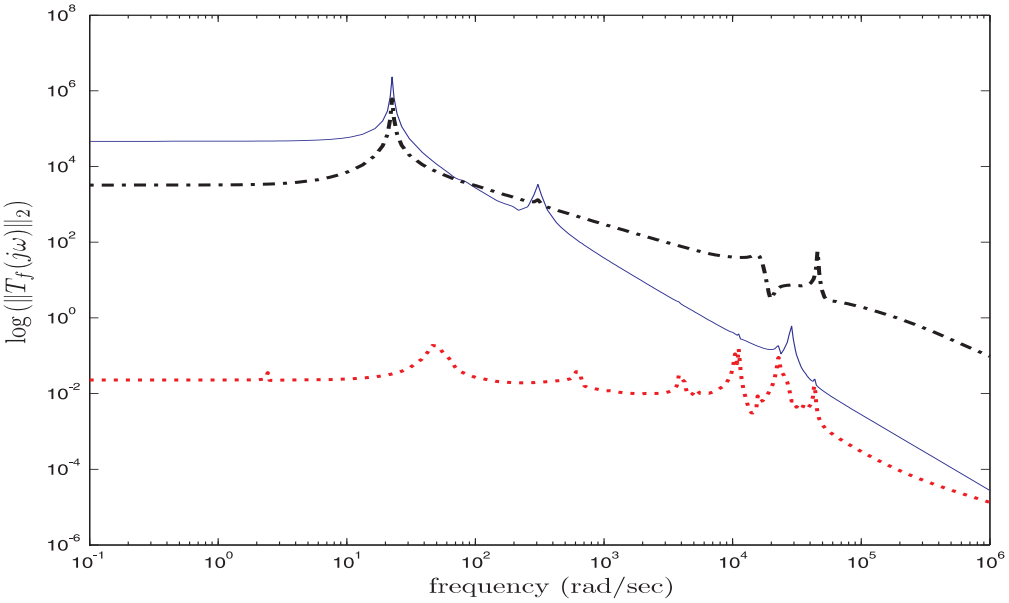


Fig. 4.19. $\sigma_{\max}$-plot of the frequency responses for CDplayer model.
— full model, ··· BT error system, -·- MLRS error system.

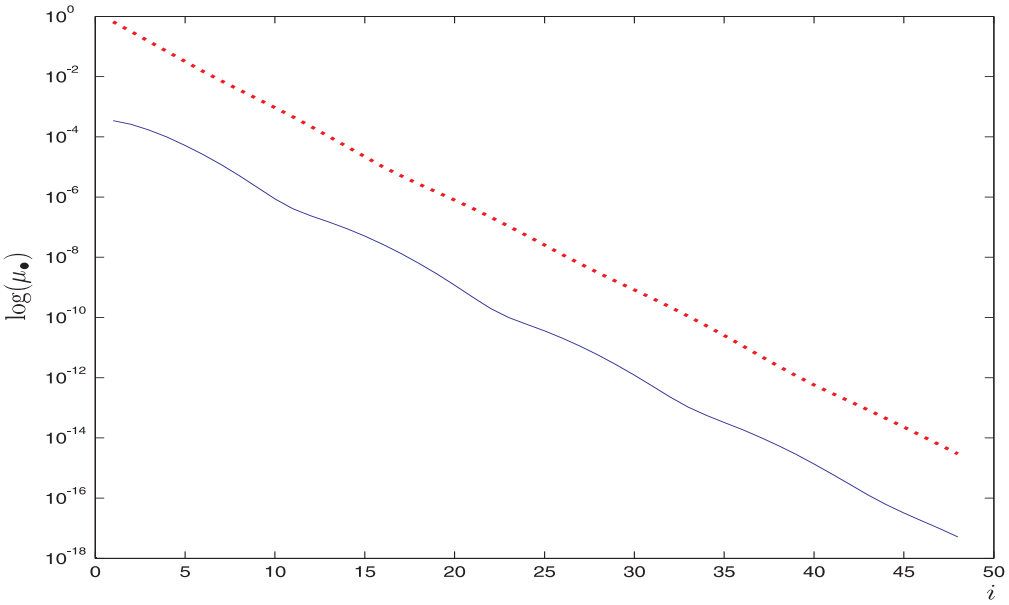


Fig. 4.20. Evolution of the values of $\mu_\bullet$ for CDplayer model.
— μ_c and ··· μ_o.

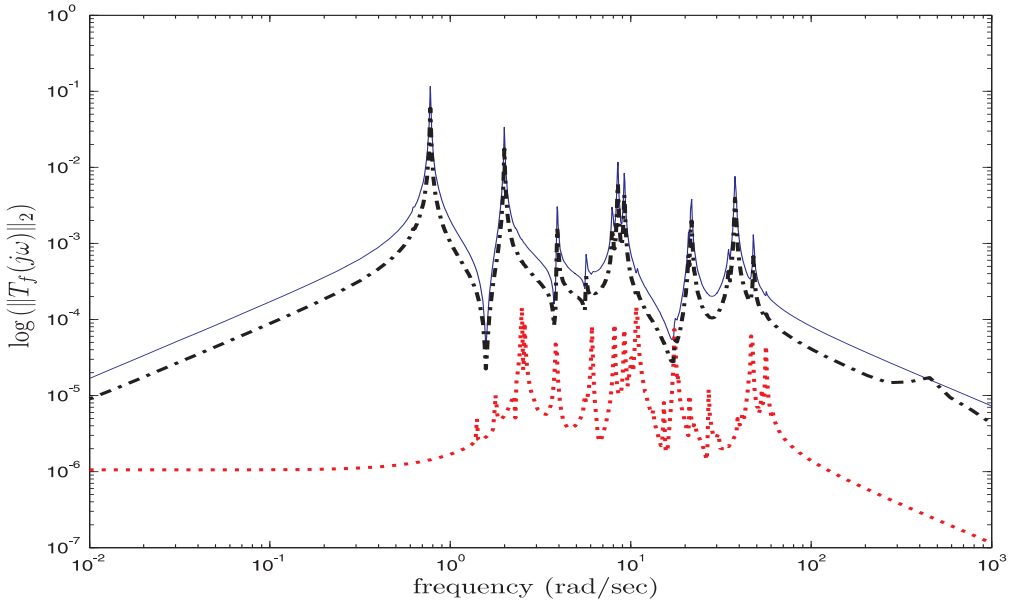


Fig. 4.21. $\sigma_{\max}$-plot of the frequency responses for ISS 1R model.
— full model, ··· BT error system, -·- MLRS error system.

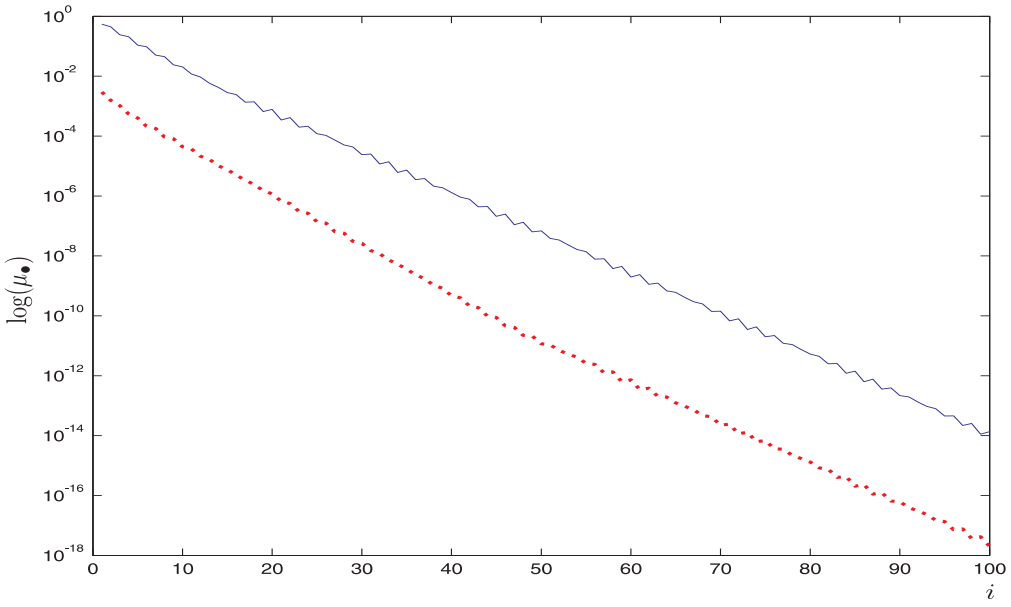


Fig. 4.22. Evolution of the values of $\mu_\bullet$ for ISS 1R model.
— μ_c and ··· μ_o.

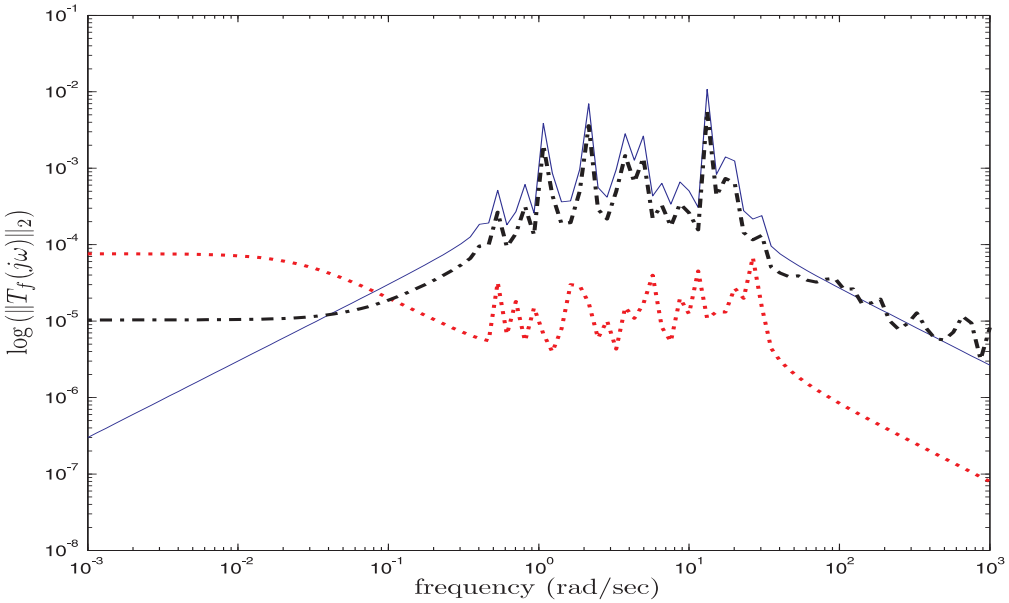


Fig. 4.23. $\sigma_{\max}$-plot of the frequency responses for ISS 12A model.
— full model, ··· BT error system, -·- MLRS error system.

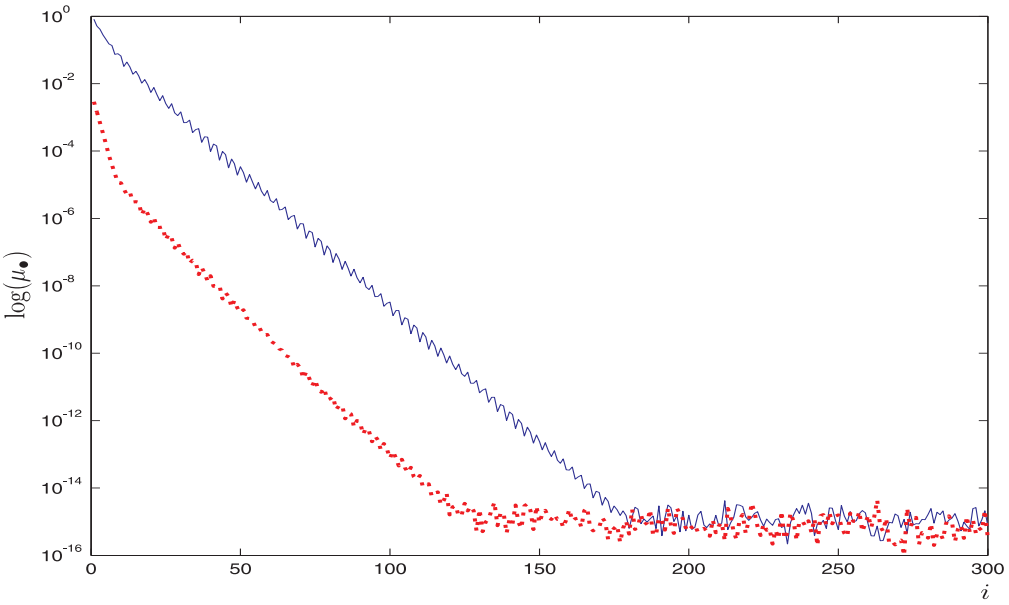


Fig. 4.24. Evolution of the values of $\mu_\bullet$ for ISS 12A model.
— μ_c and ··· μ_o.

4.9 Concluding remarks

In this chapter we presented an analysis of an efficient incremental algorithm to compute the dominant subspace of a given matrix M. Although similar algorithms have been discussed in the literature [40], we have given here a more efficient implementation along with a fairly tight bound on its accuracy and estimators that can be used in practice to monitor that accuracy.

The main contributions are the following :

- A CGS-like algorithm of complexity close to $8Nmn$ flops was derived for computing a rank n approximation of an $N \times m$ matrix M.
- A posteriori bounds for the accuracy of the approximation error were presented and their reliability was illustrated.
- The effect of round-off was studied and it was shown that the algorithm behaves much better than what can be expected for CGS. An explanation of this phenomenon was given and illustrated by numerical experiments. The effect of propagation of round-off errors was also analyzed and shown to be negligible for the applications considered up to now.

We showed also how to use the resulting approximations to provide an Approximated Balanced Truncation reduced model.
Applications to image reconstruction and model reduction show that the quality of the approximation depends crucially of the gap between the kept part and the neglected one. So one has to choose efficiently the order to obtain a good result. But despite this drawback, the algorithm has a good computational performance.

Chapter 5

Recursive Low-Rank Gramian approximation (RLRG)

In the previous chapter, we have presented a new recursive scheme to approximate the "square root" of a low numerical rank Gramian. The updating algorithm was shown to be numerically stable and faster than the approach based on the exact "square root". We showed also how to use this scheme to provide a reduced order model.
The comparison with a non-adaptive SVD scheme for image sequences shows that this algorithm achieves similar accuracy levels for image reconstruction at a significantly lower computational cost. But the comparison with Balanced Truncation for model reduction shows that this algorithm depends a lot on the size of the gap between what we keep and what we neglect. The key idea of our modified low-rank Smith method was based on the fact that the "square root" of Gramians (say e.g. controllability Gramian) can be constructed iteratively using

$$\begin{cases} \mathcal{C}_1 &= B \\ \mathcal{C}_{i+1} &= \left[\mathcal{C}_i \; A^i B\right] \end{cases}.$$

But, this square root can also be constructed in two different ways [48]. One can rewrite :

$$\mathcal{C}_{i+1} = \left[B \; AB \; \ldots \; A^{i-1}B \; A^i B\right],$$

as

$$\mathcal{C}_{i+1} = \left[\left[B \; AB \; \ldots \; A^{i-1}B\right] \; A^i B\right] = \left[\mathcal{C}_i \; A^i B\right],$$

which lead to the idea of the modified low-rank Smith algorithm. A second manner is to write it as

$$\begin{aligned} \mathcal{C}_{i+1} &= \left[B \; \left[AB \; \ldots \; A^{i-1}B \; A^i B\right]\right] \\ &= \left[B \; A\left[B \; \ldots \; A^{i-2}B \; A^{i-1}B\right]\right] \\ &= \left[B \; A\mathcal{C}_i\right]. \end{aligned}$$

If one has a good low-rank approximation of $\mathcal{C}_i$ we will have also a good low-rank approximation of $\mathcal{C}_{i+1}$ using those formulas. This formulation leads to a new scheme. It is also a low-rank Gramian method, called the *Recursive Low-Rank Gramian* (*RLRG*). This approach has the important property that it can be generalized to time-varying systems as well, unlike the previous method.
In this chapter, we present this new algorithm. We derive bounds for the approximation error and illustrate its efficiency on a few numerical examples.

5.1 RLRG approximations

As mentioned earlier, when model reduction is to be used, the Gramians have often rapidly decaying eigenvalues [98, 7], which suggests to approximate the Gramians at each step by a low-rank factorization.

We show below how to obtain such approximations for the time-varying case and at the same time exploit the sparsity of the model $\{A_\bullet, B_\bullet, C_\bullet\}$ if there is.

Recall first that for a time-varying system, we consider a time window $[k_i, k_f]$ which we consider as our time domain. Also the solutions of

$$\mathcal{G}_c(i+1) = A_i \mathcal{G}_c(i) A_i^* + B_i B_i^*, \tag{5.1}$$

and

$$\mathcal{G}_o(i) = A_i^* \mathcal{G}_o(i+1) A_i + C_i^* C_i, \tag{5.2}$$

are always symmetric positive semi-definite (§.1.2.7), so we can substitute them by symmetric factorizations

$$\mathcal{G}_c(i) = \mathcal{C}_i \mathcal{C}_i^*, \quad \text{and} \quad \mathcal{G}_o(i) = \mathcal{O}_i \mathcal{O}_i^*.$$

The key idea of the low-rank method is to approximate the factors of $\mathcal{G}_c(i)$ and $\mathcal{G}_o(i)$, $\mathcal{C}_i$ and $\mathcal{O}_i$ by their rank n_i approximations $S_n(i)$ and $R_n(i)$, respectively, at each iteration (typically n_i is constant, i.e., $n_i = n$). Note that (5.1) and (5.2) evolve differently with time (§.1.2.7), so we have to pay attention to this in the algorithm. The proposed algorithm is the following :

Algorithm 6 The Recursive Low-Rank Gramians algorithm (RLRG).

- Initialization

$$S_n(k_i) = 0, \quad \text{and} \quad R_n(k_f) = 0.$$

- The i^{th} low-rank approximations $S_n(k_i + i)$ and $R_n(k_f - i)$ are obtained as follows.
  Compute the following short singular value decompositions (SSVD) :

$$\hat{S}(k_i + i) = \left[\, B_{k_i+i-1} \middle| A_{k_i+i-1} S_n(k_i + i - 1) \,\right] = U_c \Sigma_c V_c^T, \tag{5.3}$$

$$\hat{R}(k_f - i) = \left[\, C_{k_f-i}^T \middle| A_{k_f-i}^T R_n(k_f - i + 1) \,\right] = U_o \Sigma_o V_o^T. \tag{5.4}$$

  Construct

$$\left[\, S_n(k_i + i) \middle| E_c(k_i + i) \,\right] = \underbrace{\left[\, B_{k_i+i-1} \middle| A_{k_i+i-1} S_n(k_i + i - 1) \,\right]}_{\hat{S}(k_i+i)} \left[\, V_c^{(1)} \middle| V_c^{(2)} \,\right], \tag{5.5}$$

$$\left[\, R_n(k_f - i) \middle| E_o(k_f - i) \,\right] = \underbrace{\left[\, C_{k_f-i}^T \middle| A_{k_f-i}^T R_n(k_f - i + 1) \,\right]}_{\hat{R}(k_f-i)} \left[\, V_o^{(1)} \middle| V_o^{(2)} \,\right], \tag{5.6}$$

  where $V_c^{(1)} \in \mathbb{R}^{(m+n)\times n}$ and $V_o^{(1)} \in \mathbb{R}^{(p+n)\times n}$ come from the SSVDs (5.3) and (5.4), and $E_c(k_i + i)$ and $E_o(k_f - i)$ are neglected at each iteration i.

To simplify the notations we consider :

$$l = k_i + i, \quad \text{and} \quad r = k_f - i.$$

It is immediate from the previous algorithm that

$$\mathcal{P}_i = S_n(l) S_n(l)^T, \quad \text{and} \quad \mathcal{Q}_i = R_n(r) R_n(r)^T$$

are the best rank n approximations to $\hat{S}(l)\hat{S}(l)^T$ and $\hat{R}(r)\hat{R}(r)^T$, respectively.
But this is not sufficient since we want to compare $\mathcal{P}_i$ and $\mathcal{Q}_i$ with $\mathcal{G}_c(l) = \mathcal{C}_l \mathcal{C}_l^T$ and $\mathcal{G}_o(r) = \mathcal{O}_r \mathcal{O}_r^T$, respectively. This is analyzed below. For this, we define

$$\mathcal{C}_l = \left[\, B_{l-1} \middle| A_{l-1}B_{l-2} \middle| \ldots \middle| \Phi(l,k_i)\mathcal{C}_{k_i} \,\right], \tag{5.7}$$

and

$$\mathcal{O}_r = \left[\, B_r \middle| A_r B_{r+1} \middle| \ldots \middle| \Phi(k_f,r)\mathcal{O}_{k_f} \,\right]. \tag{5.8}$$

We have the following result :

Theorem 5.1.
At each iteration, there exists orthogonal matrices

$$V_c^{(i)} \in \mathbb{R}^{(n+im)\times(n+im)} \quad \textit{and} \quad V_o^{(i)} \in \mathbb{R}^{(n+ip)\times(n+ip)},$$

satisfying :

$$\mathcal{C}_l V_c^{(i)} = \left[\, S_n(l) \middle| E_c(l) \middle| A_{l-1}E_c(l-1) \middle| \ldots \middle| \Phi(l,k_i)E_c(k_i) \,\right],$$

and

$$\mathcal{O}_r V_o^{(i)} = \left[\, R_n(r) \middle| E_c(r) \middle| A_r^T E_c(r+1) \middle| \ldots \middle| \Phi(k_f,r)^T E_c(k_f) \,\right],$$

where $E_c(i)$ and $E_o(i)$ are the neglected parts at iteration i (5.5 and 5.6). ■

Proof. We just show the proof for $V_c^{(i)}$, the other is similar.
At each step, there exists an orthogonal matrix V_c (5.5) such that

$$\left[\, B_{l-1} \middle| A_{l-1}S_n(l-1) \,\right] V_c = \left[\, S_n(l) \middle| E_c(l) \,\right].$$

For $k = 0$ we have $\mathcal{C}_{k_i} = \left[\, S_n(k_i) \middle| E_c(k_i) \,\right]$. We prove the general result by induction. Suppose that there exists an orthogonal matrix $V_c^{(i)}$ such that

$$\mathcal{C}_l V_c^{(i)} = \left[\, S_n(l) \middle| E_c(l) \middle| A_{l-1}E_c(l-1) \middle| \ldots \middle| \Phi(l,k_i)E_c(k_i) \,\right].$$

Since $\mathcal{C}_{l+1}$ can be obtained from $\mathcal{C}_l$ (5.7) as follows

$$\mathcal{C}_{l+1} = \left[\, B_l \middle| A_l\mathcal{C}_l \,\right],$$

we choose

$$V_c^{(i)} = \begin{bmatrix} I_m & 0 \\ 0 & V_c^{(i)} \end{bmatrix} \begin{bmatrix} V_c & 0 \\ 0 & I_{(i+1)m} \end{bmatrix},$$

from which it follows that

$$\begin{aligned}
\mathcal{C}_{l+1}V_c^{(i+1)} &= \left[\, B_l \middle| A_l\mathcal{C}_l \,\right] \begin{bmatrix} I_m & 0 \\ 0 & V_c^{(i)} \end{bmatrix} \begin{bmatrix} V_c & 0 \\ 0 & I_{(i+1)m} \end{bmatrix} \\
&= \left[\, B_l \middle| A_l\mathcal{C}_l V_c^{(i)} \,\right] \begin{bmatrix} V_c & 0 \\ 0 & I_{(i+1)m} \end{bmatrix} \\
&= \left[\, B_l \middle| A_l S_n(l) \middle| A_l E_c(l) \middle| \ldots \middle| \Phi(l+1,k_i)E_c(k_i) \,\right] \begin{bmatrix} V_c & 0 \\ 0 & I_{(i+1)m} \end{bmatrix} \\
&= \left[\, S_n(l+1) \middle| E_c(l+1) \middle| A_l E_c(l) \middle| \ldots \middle| \Phi(l+1,k_i)E_c(k_i) \,\right].
\end{aligned}$$

□

We can use this result to compare $\mathcal{G}_c(l)$ and $\mathcal{G}_o(r)$ with $\mathcal{P}_i$ and $\mathcal{Q}_i$, respectively, as follows :

$$\begin{aligned}\mathcal{G}_c(l) &= \mathcal{C}_l\mathcal{C}_l^T \\ &= \underbrace{S_n(l)S_n(l)^T}_{\mathcal{P}_i} + E_c(l)E_c(l)^T + \sum_{j=0}^{i-1} \Phi(l,k_i+j)E_c(k_i+j)E_c(k_i+j)^T\Phi(l,k_i+j)^T.\end{aligned}$$

Since $\Phi(l,l) = I_N$ it follows that

$$\mathcal{G}_c(l) = \mathcal{P}_i + \sum_{j=0}^{i} \Phi(l,k_i+j)E_c(k_i+j)E_c(k_i+j)^T\Phi(l,k_i+j)^T. \tag{5.9}$$

Similarly we have

$$\mathcal{G}_o(r) = \mathcal{Q}_i + \sum_{j=0}^{i} \Phi(k_f-j,r)^T E_c(k_f-j)^T E_c(k_f-j)\Phi(k_f-j,r). \tag{5.10}$$

Taking norms we obtain

$$\|\mathcal{G}_c(l) - \mathcal{P}_i\|_2 \leq \sum_{j=0}^{i} \|\Phi(l,k_i+j)\|_2^2 \|E_c(k_i+j)\|_2^2,$$

and

$$\|\mathcal{G}_o(r) - \mathcal{Q}_i\|_2 \leq \sum_{j=0}^{i} \|\Phi(k_f-j,r)\|_2^2 \|E_c(k_f-j)\|_2^2.$$

Now, define

$$\eta_c = \max_{k_i \leq l \leq \infty} \|E_c(l)\|_2, \quad \text{and} \quad \eta_o = \max_{-\infty \leq r \leq k_f} \|E_o(r)\|_2.$$

If we suppose that our system is asymptotically stable, we can bound the differences between $\mathcal{P}_i$ and $\mathcal{G}_c(l)$ and between $\mathcal{Q}_i$ and $\mathcal{G}_o(r)$ for all i,

$$\mathcal{E}_c(i) \doteq \mathcal{G}_c(l) - \mathcal{P}_i, \quad \text{and} \quad \mathcal{E}_o(i) \doteq \mathcal{G}_o(r) - \mathcal{Q}_i,$$

in terms of the "noise" levels η_c and η_o as follows :

Theorem 5.2.

If moreover the system is stable, i.e.,

$$\|\Phi(k,k_0)\| \leq m \cdot a^{(k-k_0)}, \text{ with } m > 0,\ 0 < a < 1.$$

Then

$$\|\mathcal{E}_c(i)\|_2 \leq \eta_c^2 m^2 \sum_{j=1}^{i} a^{2j} \leq \frac{\eta_c^2 m^2}{1-a^2}, \quad \textit{and} \quad \|\mathcal{E}_o(i)\|_2 \leq \eta_o^2 m^2 \sum_{j=1}^{i} a^{2j} \leq \frac{\eta_c^2 m^2}{1-a^2},$$

■

5.2 Time-invariant case

For linear time-invariant systems $\{A, B, C\}$, the differences $\mathcal{E}_c(i)$ and $\mathcal{E}_o(i)$ remain bounded for large i. We then have the following result.

Theorem 5.3.
Let $\mathcal{P}$ and $\mathcal{Q}$ be the solutions of

$$\mathcal{P} = A\mathcal{P}A^T + I, \quad \textit{and} \quad \mathcal{Q} = A^T\mathcal{Q}A + I.$$

Define

$$\eta_c = \max_{k_0 \le i \le \infty} \|E_c(i)\|_2, \quad \textit{and} \quad \eta_o = \max_{k_0 \le i \le \infty} \|E_o(i)\|_2$$

Then

$$\|\mathcal{E}_c(i)\|_2 \le \eta_c^2 \|\mathcal{P}\|_2 \le \eta_c^2 \frac{\kappa(A)^2}{1-\rho(A)^2}, \tag{5.11}$$

and

$$\|\mathcal{E}_o(i)\|_2 \le \eta_o^2 \|\mathcal{Q}\|_2 \le \eta_o^2 \frac{\kappa(A)^2}{1-\rho(A)^2}, \tag{5.12}$$

where $\kappa(A)$ and $\rho(A)$ are respectively the condition number and the spectral radius of A. ■

Proof. Here also we show only the bound for $\mathcal{E}_c(i)$, the second one can be shown similarly. It follows from (5.9) that

$$\mathcal{E}_c(i+1) = A\mathcal{E}_c(i)A^T + E_c(i)E_c(i)^T.$$

With

$$\eta_c = \max_{k_0 \le i \le \infty} \|E_c(i)\|_2,$$

we can consider the equation :

$$\mathcal{X}_{i+1} = A\mathcal{X}_i A^T + (\eta_c^2 I_N - E_c(i)E_c(i)^T), \quad \mathcal{X}_0 = 0.$$

Its iterates $\mathcal{X}_i$ are clearly positive semi-definite and hence converge to a solution $\mathcal{X}$ which is also positive semi-definite.
Moreover by linearity we have

$$\mathcal{E}_c(i+1) + \mathcal{X}_{i+1} = A(\mathcal{E}_c(i) + \mathcal{X}_i)A^T + \eta_c^2 I_N.$$

It then follows that

$$\lim_{i \to \infty} \mathcal{E}_c(i) + \mathcal{X}_i = \eta_c^2 \mathcal{P},$$

and we obtain

$$\|\mathcal{E}_c(i)\|_2 \le \eta_c^2 \|\mathcal{P}\|_2.$$

The second bound follows from the eigen-decomposition of A. □

We also have the following result on the quality of the approximation of the product of the Gramians.

Theorem 5.4.
Let $\mathcal{P}$ and $\mathcal{Q}$ be the solutions of

$$\mathcal{P} = A\mathcal{P}A^T + I, \quad \textit{and} \quad \mathcal{Q} = A^T\mathcal{Q}A + I.$$

Define

$$\eta_c = \max_{k_0 \le i \le \infty} \|E_c(i)\|_2, \quad \textit{and} \quad \eta_o = \max_{k_0 \le i \le \infty} \|E_o(i)\|_2$$

Then

$$\|\mathcal{G}_c\mathcal{G}_o - \mathcal{P}\mathcal{Q}\|_2 \le \frac{\kappa(A)^2}{1-\rho(A)^2}\left(\eta_c^2\|\mathcal{G}_o\|_2 + \eta_o^2\|\mathcal{G}_c\|_2\right), \tag{5.13}$$

where $\kappa(A)$ and $\rho(A)$ are respectively the condition number and the spectral radius of A. ■

Proof. We can consider the equation

$$\mathcal{G}_c\mathcal{G}_o - \mathcal{P}\mathcal{Q} = (\mathcal{G}_c - \mathcal{P})\mathcal{G}_o + \mathcal{P}(\mathcal{G}_o - \mathcal{Q}).$$

Taking norm of this equation yields

$$\|\mathcal{G}_c\mathcal{G}_o - \mathcal{P}\mathcal{Q}\|_2 = \|\mathcal{G}_c - \mathcal{P}\|_2.\|\mathcal{G}_o\|_2 + \|\mathcal{P}\|_2.\|\mathcal{G}_o - \mathcal{Q}\|_2.$$

Finally, using the previous theorem and the fact that $\|\mathcal{P}\|_2$ is always bounded from above by $\|\mathcal{G}_c\|_2$, we obtain

$$\mathcal{X}_{i+1} = A\mathcal{X}_i A^T + (\eta_c^2 I_N - E_c(i)E_c(i)^T), \quad \mathcal{X}_0 = 0.$$

Its iterates $\mathcal{X}_i$ are clearly positive semi-definite and hence converge to a solution $\mathcal{X}$ which is also positive semi-definite.
Moreover by linearity we have

$$\|\mathcal{G}_c\mathcal{G}_o - \mathcal{P}\mathcal{Q}\|_2 \le \frac{\kappa(A)^2}{1-\rho(A)^2}\left(\eta_c^2\|\mathcal{G}_o\|_2 + \eta_o^2\|\mathcal{G}_c\|_2\right).$$

□

This result says that if one Gramian is not well approximated, the product of the Gramians, which is related to the Hankel singular values, will not be well approximated.

Remark 5.5.

- Our bounds are very similar to those obtained in [67] for the time-invariant case; In (5.11) and (5.12), $\kappa(A)^2/(1-\rho(A)^2)$ is constant and it is very small when $\rho(A) \ll 1$ and $\kappa(A)$ is reasonable;
- η_c and η_o can be taken equal to the maximum of $\|E_c(i)\|_2$ and $\|E_o(i)\|_2$, respectively, for $k_i \le i \le \infty$, since we can interpret Theorems 5.1 and 5.2 as starting with any interval $[k_i, k_f]$. This is particulary useful if after step k_0 the errors have converged to their minimal value, i.e., the convergence threshold ϵ_m. In fact, η_c and η_o are function of k_0 and one can write

  $$\eta_c(k_0) = \max_{k_0 \le i \le \infty} \|E_c(i)\|_2, \quad \text{and} \quad \eta_o(k_0) = \max_{-\infty \le i \le k_0} \|E_o(i)\|_2.$$

  Since $\eta_c(k_0)$ and $\eta_o(k_0)$ are typically decreasing we can replace it by the maximum over the last iteration steps;
- As stopping criterion, we can use different approaches :
  1. Maximal number of iteration steps. The iteration is stopped after a certain number $i_{\max}$ of iterations steps. But, this criterion can be avoided by setting this $i_{\max} = \infty$. Obviously, no additional computations need to be performed to evaluate it. The drawback of this stopping criterion is that it is not related to the attainable accuracy of the delivered low-rank Gramian.

2. Stagnation of the angles. The iteration is stopped when stagnation of $\sphericalangle(S_n(i-1), S_n(i))$ and $\sphericalangle(R_n(i-1), R_n(i))$ are detected. Roughly speaking, these angles are considered as "stagnating", when no noticeable decrease is observed in 10 consecutive iteration steps. This criterion works well in practice. It requires the computation of a SVD which gives the cosine of these angles.
3. Smallness of the values η_c and η_o. For this, we predefine a tolerance ϵ_m and test if $\eta_c \leq \epsilon_m$ and $\eta_o \leq \epsilon_m$ for several iterations. Loosely speaking, this means the following. When η_c and η_o and consequently $\|E_c(i)\|$ and $\|E_o(i)\|$ become smaller than ϵ_m , then the "contribution" from following iteration is not needed for approximation.

In general, the two last criteria are affected by round-off errors, this is why there is a delay before stopping of the algorithm. Note that the delay between stagnation and stopping of the algorithm can be changed.

- At each iteration, we need to compute only $U_c^{(1)} = U_c(:, 1:n)$, $U_o^{(1)} = U_o(:, 1:n)$, η_c and η_o. This requires typically $O(N(n+m)^2)$ flops and $O(N(n+p)^2)$ flops, respectively [63], which is more reasonable (when $N \gg m, n$) than the complexity of a traditional method which has in general a complexity of $O(N^3)$ flops.

5.2.1 Convergence of the RLRG algorithm for LTI systems

In this subsection we analyze the convergence of the Recursive Low-Rank Gramian (RLRG) approximation algorithm for a linear time invariant system $\{A, B, C\}$. The convergence will allow us to deduce important results, for example for periodic time-varying systems (see §.5.3).
Although all material below applies to both approximations $S_n(\bullet)$ and $R_n(\bullet)$, we focus on the controllability version only, $S_n(\bullet)$.

First, note that the updating transformation for $S_n(\bullet)$ (5.3 and 5.5) is nonlinear and implicit. Thus to prove convergence of the RLRG algorithm, we will use a generalization of the *fixed point theorem*, due to Ortega and Reinboldt [94], called the *Contraction mapping theorem*.

Definition 5.6.
A linear operator Υ is nonexpansive if $\rho(\Upsilon) \leq 1$, and contractive if $\rho(\Upsilon) < 1$.
Note that a contractive operator is strictly nonexpansive.

Theorem 5.7.
The nonlinear system $S_+ = f(S)$[1] admits a fixed point S_f iff there exists a contractive linear operator ∇f so that for all S we have

$$f(S_f + tS) = f(S_f) + t\nabla f.S + O(t^2).$$

∇f is called Gâteaux-derivative of f. ■

Notice that if the linear operator ∇f is only nonexpansive we have a fixed *region* or *set* rather than a fixed *point*. Now, how to apply this result to our case?

Firstly, for the RLRG algorithm, it is obvious that the differentiability depends on the differentiability of the SVD which is guaranteed if there is a gap between the part that we keep and the part that we neglect in (5.3) and (5.5) [63], and this is supposed to be the case. To prove the convergence we thus have to prove that the updating mapping (5.5) is contractive. For this, let us consider a perturbation of $S = S_n$, namely $S + \Delta$.
From (5.3) we can define the SVD

$$\left[\, A\mathcal{S} \middle| B \,\right] = U \begin{bmatrix} \Sigma_+ & 0 \\ 0 & \Sigma_- \\ 0 & 0 \end{bmatrix} V^T,$$

[1] subscript + to emphasize the updated version of S.

and using these U and V matrices, we have :

$$\left[\, A\Delta \middle| 0 \,\right] = U\hat{\Delta}V^T \qquad \text{where} \qquad \hat{\Delta} = \begin{bmatrix} \hat{\Delta}_{11} & \hat{\Delta}_{12} \\ \hat{\Delta}_{21} & \hat{\Delta}_{22} \\ \hat{\Delta}_{31} & \hat{\Delta}_{32} \end{bmatrix}, \tag{5.14}$$

is partitioned conformably with Σ. Let us consider the partitioned transformations

$$V = \left[\, V_1 \middle| V_2 \,\right] = \left[\begin{array}{c|c} V_{11} & V_{12} \\ \hline V_{21} & V_{22} \end{array}\right], \qquad \text{and} \qquad U = \left[\, U_1 \middle| U_2 \middle| U_3 \,\right], \tag{5.15}$$

and define $\tilde{\Sigma}_+ \doteq \Sigma_+ + \Delta_{11}$ and $\tilde{\Sigma}_- \doteq \Sigma_- + \Delta_{22}$.
We can distinguish two cases : V constant and V varying.
If V is still constant then the new version of S is given by

$$\mathcal{S}_+ = \left[\, A\mathcal{S} \middle| B \,\right] V_1 = U \begin{bmatrix} \Sigma_+ \\ 0 \\ 0 \end{bmatrix}$$

and the perturbed version of S_+ is given by

$$\mathcal{S}_+ + \Delta_+ = \left[\, A(\mathcal{S}+\Delta) \middle| B \,\right] V_1 = U \begin{bmatrix} \tilde{\Sigma}_+ \\ \hat{\Delta}_{21} \\ \hat{\Delta}_{31} \end{bmatrix},$$

and thus

$$\Delta_+ = \left[\, A\Delta \middle| 0 \,\right] V_1 = A\Delta V_{11}.$$

Using the *vec* formulation (§.1.1.4) we obtain

$$vec(\Delta_+) = (V_{11}^T \otimes A).vec(\Delta).$$

Here, the term $V_{11}^T \otimes A$ corresponds to the linear operator ∇f of Theorem 5.7.
As

$$\rho(V_{11}^T \otimes A) = \rho(V_{11}).\rho(A) < 1$$

the mapping $\Delta \to \Delta_+$ is a contraction.

Let now V be varying as well. The new version of S is still given by

$$\mathcal{S}_+ = \left[\, A\mathcal{S} \middle| B \,\right] V_1 = U_1\Sigma_+,$$

and the perturbed version is given by

$$\mathcal{S}_+ + \Delta_+ = \left[\, A(\mathcal{S}+\Delta) \middle| B \,\right] V_1(\Delta) = U_1(\Delta)\hat{\Sigma}_+,$$

and so

$$\Delta_+ = U_1(\Delta)\hat{\Sigma}_+ - U_1\Sigma_+.$$

Let write the transformation $U(\Delta)$ as follows

$$U(\Delta) = U \begin{bmatrix} I & -Q^T \\ Q & I \end{bmatrix} + O(\|\Delta\|_2^2),$$

then Q can be solved in first order approximation [116] (see also Theorem 4.1, Page 57) from

$$\begin{bmatrix} I & Q^T \\ -Q & I \end{bmatrix} \begin{bmatrix} \tilde{\Sigma}_+\tilde{\Sigma}_+^T & \tilde{\Sigma}_+\hat{\Delta}_{21}^T + \hat{\Delta}_{12}\tilde{\Sigma}_- & \tilde{\Sigma}_+\hat{\Delta}_{31}^T \\ \hat{\Delta}_{21}\tilde{\Sigma}_+ + \tilde{\Sigma}_-\hat{\Delta}_{12}^T & \tilde{\Sigma}_-\tilde{\Sigma}_-^T & \Sigma_-\hat{\Delta}_{32}^T \\ \hat{\Delta}_{31}\tilde{\Sigma}_+ & \hat{\Delta}_{32}\tilde{\Sigma}_- & 0 \end{bmatrix} \begin{bmatrix} I & -Q^T \\ Q & I \end{bmatrix} + O(\|\Delta\|_2^2)$$

$$= \begin{bmatrix} \hat{\Sigma}_+^2 & 0 & 0 \\ 0 & \hat{\Sigma}_-^2 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

Now, if we consider the $(2:3,1)$ blocks, we have

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} = -Q(\tilde{\Sigma}_+\tilde{\Sigma}_+^T) + \begin{bmatrix} \tilde{\Sigma}_-\tilde{\Sigma}_-^T & \tilde{\Sigma}_-\hat{\Delta}_{32}^T \\ \hat{\Delta}_{32}\tilde{\Sigma}_- & 0 \end{bmatrix} Q + \begin{bmatrix} \hat{\Delta}_{21}\tilde{\Sigma}_+ + \tilde{\Sigma}_-\hat{\Delta}_{12}^T \\ \hat{\Delta}_{31}\tilde{\Sigma}_+ \end{bmatrix} + O(\|\Delta\|_2^2).$$

This equation can be solved in first order [115], and if we neglect Σ_- versus Σ_+ (i.e. $\|\Sigma_+^{-1}\|_2.\|\Sigma_-\|_2 \simeq O(\|\Delta\|_2))^2$, we obtain

$$\|Q - \begin{bmatrix} \hat{\Delta}_{21}\tilde{\Sigma}_+^{-1} \\ \hat{\Delta}_{31}\tilde{\Sigma}_+^{-1} \end{bmatrix}\|_2 \leq \underbrace{\|Q\|_2 \frac{\|\tilde{\Sigma}_+^{-1}\|_2^2.\|\tilde{\Sigma}_-\|_2^2}{1 - \|\tilde{\Sigma}_+^{-1}\|_2^2.\|\tilde{\Sigma}_-\|_2^2}}_{c}. \tag{5.16}$$

And thus one obtains

$$\begin{aligned} \Delta_+ &= U \begin{bmatrix} I \\ \hat{\Delta}_{21}\tilde{\Sigma}_+^{-1} \\ \hat{\Delta}_{31}\tilde{\Sigma}_+^{-1} \end{bmatrix} \tilde{\Sigma}_+ - U_1\Sigma_+ + O(c) \\ &= U_1(\Sigma_+ + \hat{\Delta}_{11}) + U_2\hat{\Delta}_{21} + U_3\hat{\Delta}_{31} - U_1\Sigma_+ + O(c) \\ &= U_1\hat{\Delta}_{11} + U_2\hat{\Delta}_{21} + U_3\hat{\Delta}_{31} + O(c). \end{aligned}$$

From (5.14) we have

$$\begin{bmatrix} \hat{\Delta}_{11} \\ \hat{\Delta}_{21} \\ \hat{\Delta}_{31} \end{bmatrix} = \begin{bmatrix} U_1^T \\ U_2^T \\ U_3^T \end{bmatrix} \left[A\Delta \middle| 0 \right] V_1 \tag{5.17}$$

so

$$\begin{aligned} \Delta_+ &= U_1U_1^TA\Delta V_{11} + U_2U_2^TA\Delta V_{11} + U_3U_3^TA\Delta V_{11} + O(c) \\ &= \underbrace{(U_1U_1^T + U_2U_2^T + U_3U_3^T)}_{I} A\Delta V_{11} + O(c). \end{aligned}$$

Therefore we have

$$\Delta_+ \simeq A\Delta V_{11} + O(c).$$

Furthermore from (5.16) and (5.17) we have

$$\|Q\|_2 \approx \|A\Delta\|_2.\|\Sigma_+^{-1}\|_2,$$

and so

[2] Note that in this case

$$\|(\tilde{\Sigma}_+\tilde{\Sigma}_+^T)^{-1}O(\|\Delta\|_2^2)\| \ll O(\|\Delta\|_2^2)$$

$$c = \|Q\|_2 \frac{\|\tilde{\Sigma}_+^{-1}\|_2^2.\|\tilde{\Sigma}_-\|_2^2}{1 - \|\tilde{\Sigma}_+^{-1}\|_2^2.\|\tilde{\Sigma}_-\|_2^2} \approx \|A\Delta\|_2 \frac{\|\tilde{\Sigma}_+^{-1}\|_2^3.\|\tilde{\Sigma}_-\|_2^2}{1 - \|\tilde{\Sigma}_+^{-1}\|_2^2.\|\tilde{\Sigma}_-\|_2^2}.$$

Using the *vec* formulation we obtain finally

$$vec(\Delta_+) = (V_{11}^T \otimes A).vec(\Delta) + O(c).$$

As

$$\rho(V_{11}^T \otimes A) = \rho(V_{11}).\rho(A) < 1$$

the mapping $\Delta \to \Delta_+$ is a contraction provided $\|\tilde{\Sigma}_+^{-1}\|_2\|\tilde{\Sigma}_-\|_2$ is sufficiently small i.e., the gap is sufficiently large. Under these conditions, the RLRG algorithm admits a fixed point. Furthermore, this fixed point has a very nice property given by the following result.

Theorem 5.8.
The fixed point of the RLRG algorithm is an $\{A, B\}$ invariant subspace provided that the matrix V_{11} (5.15) is invertible. ■

Proof.
First let us see what $\{A, B\}$ invariance means.

Lemma 5.9. *[80]*
A subspace $\mathcal{V}$ is an $\{A, B\}$ invariant subspace iff there exists a (non unique) feedback-gain matrix K such that $\mathcal{V}$ is $(A - BK)$-invariant. In other words

$$A\mathcal{V} \subset Im(B) + \mathcal{V} \quad \Leftrightarrow \quad \text{there exists } K \text{ s.t. } (A - BK)\mathcal{V} \subset \mathcal{V}.$$

■

Now let i be the iteration where we reach the fixed point, i.e.,

$$\mathrm{Im}(S_i) = \mathrm{Im}(S_{i+1}),$$

this is equivalent to say that there exists a square nonsingular matrix X s.t. $S_i X = S_{i+1}$. Then, if we put ourselves in a coordinate system where :

$$S_i = \begin{bmatrix} R \\ 0 \end{bmatrix}, \quad R \in \mathbb{R}^{n \times n},$$

(this can be obtained using for example a QR decomposition of S_i followed by a pre-multiplication of the matrix S_i by Q). The fixed singular subspace implies that we must have :

$$S_{i+1} = \begin{bmatrix} R_+ \\ 0 \end{bmatrix}, \quad R_+ \in \mathbb{R}^{n \times n}.$$

The two matrices R and R_+ are related using (5.15) as follows

$$\left[\left[\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array} \right] \left[\begin{array}{c} R \\ \hline 0 \end{array} \right] \middle| \left[\begin{array}{c} B_1 \\ \hline B_2 \end{array} \right] \right] \left[\begin{array}{c} V_{11} \\ \hline V_{21} \end{array} \right] = \left[\begin{array}{c|c} A_{11}R & B_1 \\ \hline A_{21}R & B_2 \end{array} \right] \left[\begin{array}{c} V_{11} \\ \hline V_{21} \end{array} \right] = \left[\begin{array}{c} R_+ \\ \hline 0 \end{array} \right].$$

And so, we have

$$\begin{cases} A_{11}RV_{11} + B_1V_{21} = R_+ \\ A_{21}RV_{11} + B_2V_{21} = 0 \end{cases}. \tag{5.18}$$

If V_{11} is invertible which is equivalent to $\|V_{21}\| < 1$ (this follows from the Schur complement), it follows from (5.18) that $\mathrm{Im}(S_i) = \begin{bmatrix} I \\ 0 \end{bmatrix}$ must be an $\{A, B\}$ invariant subspace since for

$$K = \left[\, K_1 \middle| 0 \,\right] = \left[\, -V_{21}V_{11}^{-1}R^{-1} \middle| 0 \,\right],$$

we have

$$A - BK = \left[\begin{array}{c|c} A_{11} - B_1K_1 & A_{12} \\ \hline A_{211} - B_2K_1 & A_{22} \end{array}\right] = \left[\begin{array}{c|c} A_{11} - B_1K_1 & A_{12} \\ \hline 0 & A_{22} \end{array}\right].$$

which concludes our proof. □

This result says that the fixed point is an $\{A, B\}$ invariant subspace, we hope that this property may help to give a complete description of the fixed points and their properties. We believe that there is a unique fixed point which is an $\{A, B\}$ invariant subspace of dimension n, but we were not able to prove this. A complete analysis of this question is by itself a new subject of research, which we leave for the future.

Remark 5.10. For the observability, we speak about $\{A^T, C^T\}$ invariance instead of $\{A, B\}$ invariance.

In the Figures 5.1, 5.2, 5.3, and 5.4 we show numerical convergence results corresponding to each of our benchmark models. Convergence is shown in terms of the cosine of the canonical angle between two successive versions of the right dominant subspaces. i.e., between V_i and V_{i-1}. The canonical angle is defined by the smallest singular value of the product $V_i^T V_{i-1}$ [63]. We show also the cosine of the angle between V_i and $\begin{bmatrix} I_n \\ 0 \end{bmatrix}$. Actually, this cosine shows the behavior of the dominant subspace. If this cosine remains close to 1, this means that we reach a fixed point and there is no change in the image of V_i even if we continue the iterations. We can see from Figures 5.1-4 and Table 5.1 that the convergence depends mainly of the spectral radius of the matrix A which is affected by numerical round off.

	building model	CDplayer model	ISS 1R model	ISS 12A model
$\rho(A)$	0.4997	1	0.7338	0.8310

Table 5.1. Spectral radius of the matrix A of each benchmark model.

For the Building and ISS 1R models the spectral radius of the matrix A (i.e., $\rho(A)$) is smaller than 1 and so we reach the fixed point rapidly (after 50 iterations for the building model which is of order 48 (Figure 5.1), and the same number of iterations for the ISS 1R model which has order 270, Figure 5.3. For the ISS 12A model, even if the spectral radius is smaller than 1, the convergence is not strict and due to round off error there is still oscillations (Figures 5.4). For the CDplayer model, the spectral radius is equal to 1 and this implies no convergence at all (Figure 5.2).

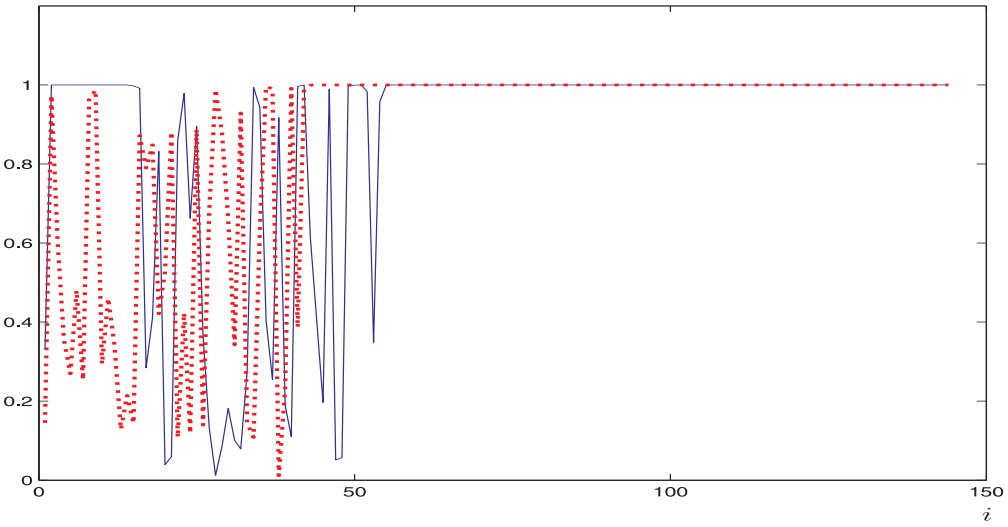


Fig. 5.1. Convergence and Cosine of subspace angles for Building model.

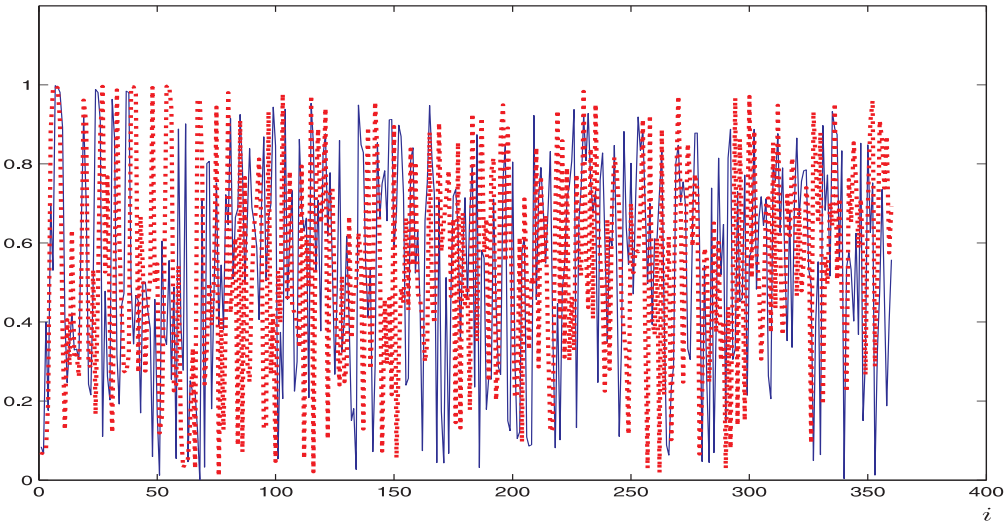


Fig. 5.2. Convergence and Cosine of subspace angles for CDplayer model.

LEGEND : — $cos(\sphericalangle(V_i, V_{i-1}))$ and $\cdots$ $cos(\sphericalangle(V_i, \begin{bmatrix} I_n \\ 0 \end{bmatrix}))$.

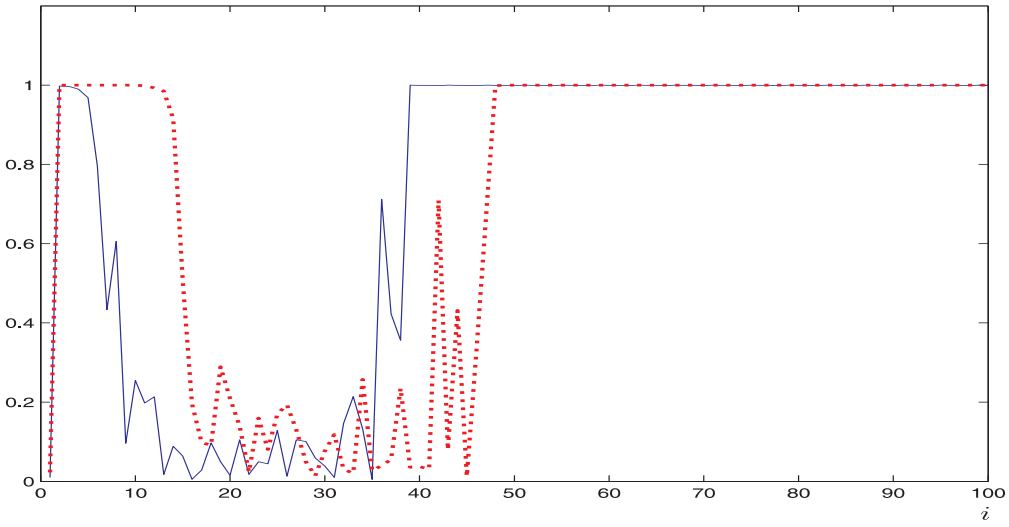


Fig. 5.3. Convergence and Cosine of subspace angles for ISS 1R model.

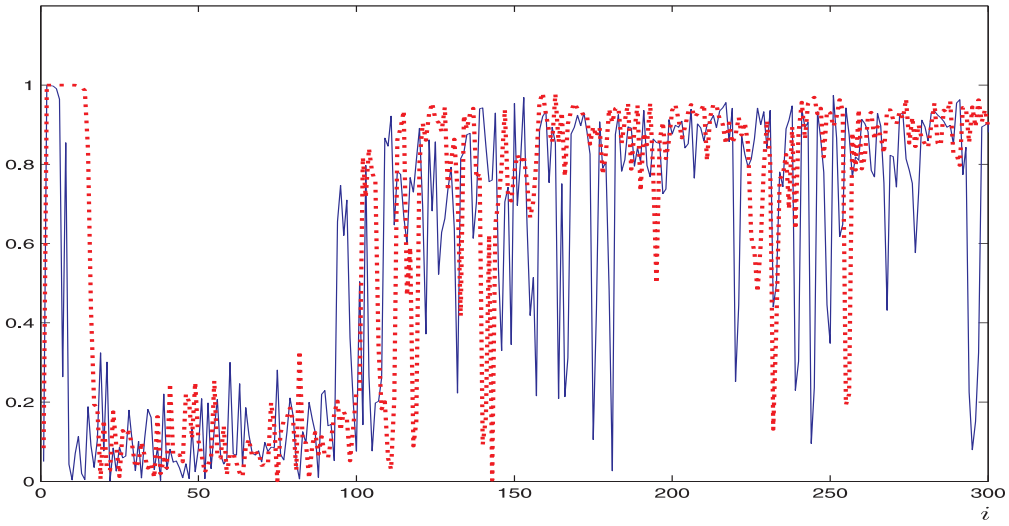


Fig. 5.4. Convergence and Cosine of subspace angles for ISS 12A model.

LEGEND : — $cos(\sphericalangle(V_i, V_{i-1}))$ and $\cdots$ $cos(\sphericalangle(V_i, \begin{bmatrix} I_n \\ 0 \end{bmatrix}))$.

5.3 Periodic case

Using the connection between the periodic time-varying system and the time-invariant system [113, 123, 121, 28] (see also Page 14), and the fact that for a periodic system there exists a periodic controllability Gramian [113], we can extend the previous time-invariant result to a K-periodic system, as follows :

Theorem 5.11.
Let $\mathcal{P}$ be the solution of $\mathcal{P} = \hat{A}\mathcal{P}\hat{A}^T + I_{KN}$ where

$$\hat{A} = \begin{pmatrix} 0 & \dots & 0 & A_0 \\ A_1 & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \vdots \\ 0 & \dots & A_{K-1} & 0 \end{pmatrix} \quad \textit{and} \quad \mathcal{P} \doteq diag(\mathcal{P}_1, \dots, \mathcal{P}_{K-1}, \mathcal{P}_0)$$

then

$$\|\mathcal{E}_c(k)\|_2 \leq \eta_c^2 \|\mathcal{P}\|_2 \leq \eta_c^2 \frac{\kappa(\hat{A})^2}{1 - \rho(\Phi(K,0))^2} \tag{5.19}$$

■

Remark 5.12. Using the connection between periodic and time-invariant systems, and using the convergence result theorem for the time-invariant case, we can conclude that we have also a periodicity for the dominant subspaces defined by the $S_n(i)$, which implies that the reduced order model will be also periodic.

5.4 RLRG versus MLRS

Even if the two algorithms have things in common (they approximate the square root of Gramians), they operate in completely different ways.
The idea of the MLRS procedure is windowing : it consists of a "window" which reviews all vectors of the matrix and collects dominant elements using a "local" SVD as a criterion of "dominance". The result will be the dominant subspace of the square root of the considered Gramian. And so the quality of the approximation depends mainly on the neglected vectors and if there is or not a considerable gap between the neglected part and the kept part.
The RLRG procedure operates completely differently. Given an initial estimate of this dominant subspace, it applies A to the old space and then performs a correction using B. This procedure also converges to the dominant subspace of the square root of the Gramian.
The principal advantage of this method is that it decreases the effect of the old approximation errors since they are weighted at each iteration by a coefficient lower than 1 (the spectral radius of A which is strictly smaller than 1 if we assume that the system is stable), and thus only the last errors dominate. Those can be made very small by adapting (in a dynamic way) the initial starting point. And so the parameters μ_c and μ_o resulting from the RLRG algorithm are in fact dominated by the last terms, i.e., $\eta_c \approx \|E_c(k)\|_2$ and $\eta_o \approx \|E_c(k)\|_2$ which will give much tighter bounds in Theorems 5.2 and 5.3.
Theorem 5.4 is still available here, but as the noise levels η_c and η_o resulting by using the Recursive Low-Rank Gramian approximation algorithm are smaller than those resulting by using the Modified Low-Rank Smith approximation algorithm, the bound is also much tighter.

The reduced order model is also computed as for the MLRS algorithm. We use the resulting low-rank approximations instead of the exact square roots in the Balanced Truncation algorithm, indeed we use the Approximated Balanced Truncation procedure to provide the reduced order model (see Algorithm 4). The results presented for the MLRS algorithm hold for the RLRG algorithm, but as the parameters $\mu_\bullet$ is smaller than those obtained for MLRS the quality of our reduced model improves with RLRG, as is shown by the numerical examples presented in the next section.

5.5 Numerical example

5.5.1 Time-varying periodic example

In this section we present a numerical example of a periodic system to which we applied the RLRG algorithm. Using the multirate sampling data model given in [121], we can construct from any discrete LTI system $\{A, B, C\}$ a periodic time-varying system which can be represented by :

$$\begin{cases} \bar{x}_{k+1} = \bar{A}_k \bar{x}_k + \bar{B}_k u_k \\ \bar{y}_k \quad = \bar{C}_k \bar{x}_k \end{cases}$$

where $\bar{x}_k^T \doteq [x_k^T \; v_k^T] \in \mathbb{R}^{N+m}$ is the enlarged state vector, and the periodic system matrices are given by

$$\bar{A}_k \doteq \begin{bmatrix} A & B(I - \Delta(k)) \\ 0 & I - \Delta(k) \end{bmatrix}, \; \bar{B}_k \doteq \begin{bmatrix} B\Delta(k) \\ \Delta(k) \end{bmatrix} \text{ and } \bar{C}_k \doteq \bar{\Delta}(k) \begin{bmatrix} C & 0 \end{bmatrix}.$$

Here $\Delta(k)$ and $\bar{\Delta}(k)$ are N-periodic $m \times m$ and $p \times p$ matrices given by

$$\Delta(k) \doteq diag\{\delta_i(k), i = 1, 2, \ldots, m\} \quad \bar{\Delta}(k) \doteq diag\{\bar{\delta}_i(k), i = 1, 2, \ldots, p\}$$
$$\delta_i(k) \doteq \begin{cases} 1, \, k = jN_i, \\ 0, \, k \neq jN_i, \end{cases} \quad j \in \mathbb{Z}^+ \qquad \delta_i(k) \doteq \begin{cases} 1, \, k = jM_i, \\ 0, \, k \neq jM_i, \end{cases} \quad j \in \mathbb{Z}^+,$$

where T is the basic period, output j is updated every M_j periods, input i is sampled every N_i periods and $N = lcm(N_i, M_j)$.

We apply this to the discretized SISO system of the arm of the CD player with a sampling time $T_s = 0.0001$, $N_i = M_i = 2$ and we reduce the order from 122 to 20. For this model we have $\frac{\kappa(\hat{A})^2}{1-\rho(\hat{A})^2} \simeq 10^5$, and $\eta \simeq 10^{-16}$, $\forall k \geq 20$.

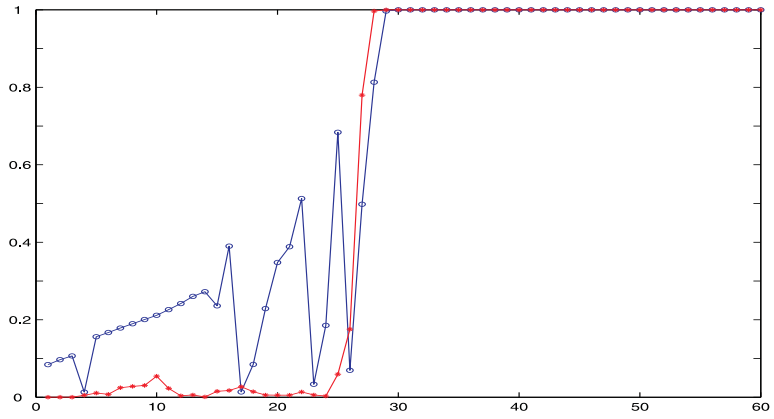
In Figures 5.5-a and -b, we show the cosine of the canonical angle between the dominant subspace of two successive iterations $(k-1)$ and k, i.e., $\cos(\sphericalangle(S_n(k-1), S_n(k)))$, and the canonical angle with the exact dominant subspace, noted $S_n(\infty)$, of the controllability Gramians $\mathcal{G}_{c_0}$ and $\mathcal{G}_{c_1}$ of the lifted LTI system (see Page 14), i.e., $\cos(\sphericalangle(S_n(k), S_n(\infty)))$.

Actually, the time-varying system obtained here is periodic of period 2. Indeed, one obtains two lifted LTI systems as we take as starting time $k = 1$ or $k = 2$. And so we have two controllability Gramians corresponding respectively to each lifted LTI system, which corresponds also to periodic Gramians for the original LTV system. Those subfigures show the convergence and the accuracy of our algorithm. It can be seen that we have convergence as soon as the algorithm has eliminated the $n = 20$ vectors of the initial matrix $S_n(0)$ for both cases.

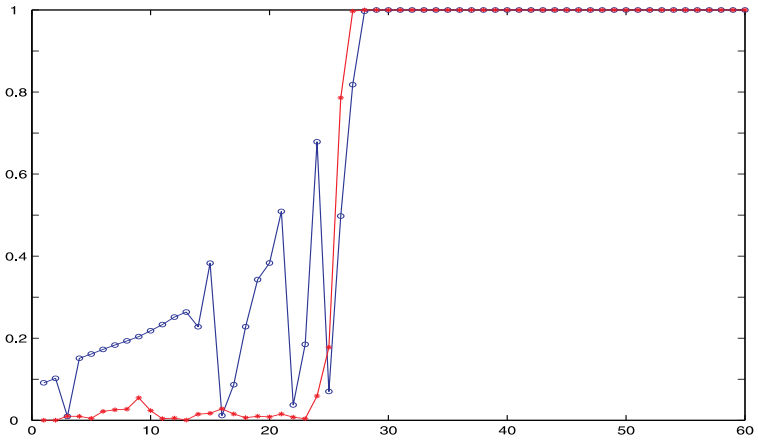
In Figure 5.6 we compare frequency responses of the time-invariant lifted systems (1.6) for starting point 1 (Figure 5.6-a) and 2 (Figure 5.6-b).

In each figure we give the amplitude of the frequency response of the original model, the absolute errors in the frequency response after 30 steps and 60 steps, and the absolute errors in the frequency response using the exact dominant subspace of the controllability matrix of the lifted LTI system. We can see that just after 60 iterations one obtains the same results as if we consider the whole matrix of controllability, but apparently the number of iterations necessary to obtain this results depend on the spectral radius of the matrix $\hat{A}$ of the lifted system.

This example shows also that we obtain a reduced-order model which is still periodic.

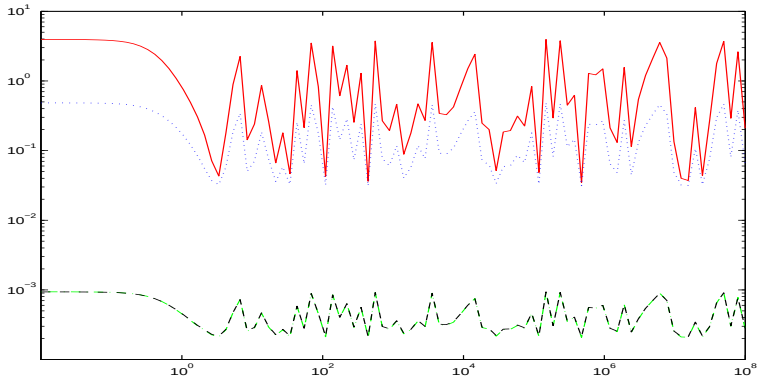


a . for starting point 1.

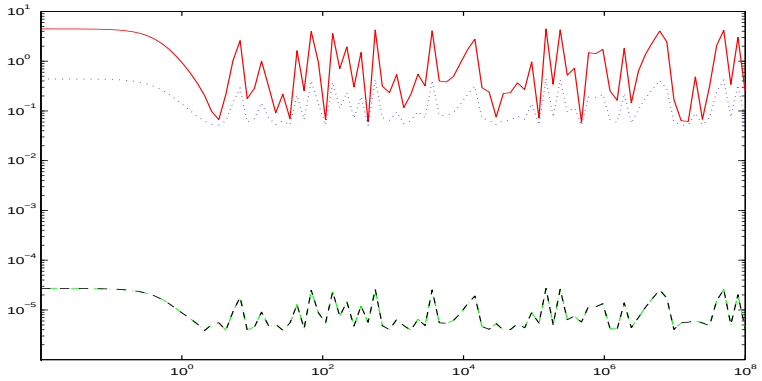


b . for starting point 2.

Fig. 5.5. Distance between dominant subspaces.
$\cos(\sphericalangle(S_n(k), S_n(k-1)))$, $\cos(\sphericalangle(S_n(k), S_n(\infty)))$



a . for starting point 1.



b . for starting point 2.

Fig. 5.6. Frequency response
— full model, ··· approx. errors (30 steps),
-·- approx. errors (60 steps), - - approx.errors (exact Gramian)

5.5.2 LTI model reduction examples

This section sums up numerical results of MLRS algorithm and gives the numerical results of the RLRG algorithm applied on the benchmark models. For each example, we show the $\mathcal{H}_\infty$ norm of the original models and the error systems in Table 5.2. Table 5.3 give a comparison of the noise level $\mu_\bullet$ corresponding to MLRS and RLRG algorithms. We tabulate also the CPU time corresponding to both algorithms in Table 5.4. σ_{max}-plot of the full order and the corresponding error systems are also shown.

It can be seen from Figures 5.7, 5.9, 5.11, and 5.13 that we obtain better results with the RLRG algorithm. These results are closer to those obtained using Balanced truncation. For the four Benchmark examples the RLRG algorithm approximates better the models than Balanced Truncation at low and medium frequencies. This is not the case for high frequencies, but the quality of the approximations are still close to those obtained using Balanced Truncation, and better than those obtained using the MLRS algorithm.

Figures 5.8, 5.10, 5.12, and 5.14 shows the noise levels. Notice that these noise levels shown must be interpreted in a special way. The noise levels must be multiplied by the corresponding power of the spectral radius of A to obtain the real values of the noise level at the end, i.e., the real noise level $\tilde{\mu}_\bullet$ is obtained as

$$\tilde{\mu}_\bullet(i) \doteq \rho(A)^{\tau-i}\mu_\bullet(i), \quad \text{where } \tau \text{ is the number of iteration.}$$

And so the values of noise levels considered in Theorems 5.2, 5.3, and 5.4 will be taken in the last obtained values which will be very small.
We notice here also that for poorly balanced systems the resulting noise levels are not of the same order as for well balanced systems. This is still the case for the CDplayer model.

model	$\|\mathcal{S}\|_\infty$	$\frac{\|\mathcal{S}-\mathcal{S}_n^{BT}\|_\infty}{\|\mathcal{S}\|_\infty}$	$\frac{\|\mathcal{S}-\mathcal{S}_n^{MLRS}\|_\infty}{\|\mathcal{S}\|_\infty}$	$\frac{\|\mathcal{S}-\mathcal{S}_n^{RLRG}\|_\infty}{\|\mathcal{S}\|_\infty}$
Building	0.0053	0.1143	0.9994	0.4301
CDplayer	$2.3198.10^{6}$	$8.0704.10^{-8}$	0.9875	$6.8931.10^{-6}$
ISS 1R	0.1159	0.0013	1.0000	0.1023
ISS 12A	0.0107	0.0071	0.9992	0.9697

Table 5.2. H_∞ norm of benchmark models, the MLRS and the RLRG error systems.

model	μ_c^{MLRS}	μ_o^{MLRS}	μ_c^{RLRG}	μ_o^{RLRG}
Building	$3.4411.10^{-4}$	0.6655	$6.5063.10^{-15}$	$4.3799.10^{-12}$
CDplayer	6.3739	6.3234	$4.0575.10^{-20}$	$6.0341.10^{-20}$
ISS 1R	0.5494	0.0030	$1.5063.10^{-8}$	$1.1641.10^{-10}$
ISS 12A	0.8248	0.0029	$7.1973.10^{-25}$	$1.1751.10^{-27}$

Table 5.3. MLRS and RLRG noise levels $\mu_\bullet$ for benchmark models.

model	BT	MLRS	RLRG
Building	0.3750	0.1720	0.3380
CDplayer	0.7970	1.8750	0.7340
ISS 1R	11.6720	9.4530	4.7350
ISS 12A	$1.1327.10^{3}$	$0.6794.10^{3}$	$0.1029.10^{3}$

Table 5.4. CPU time for different algorithms

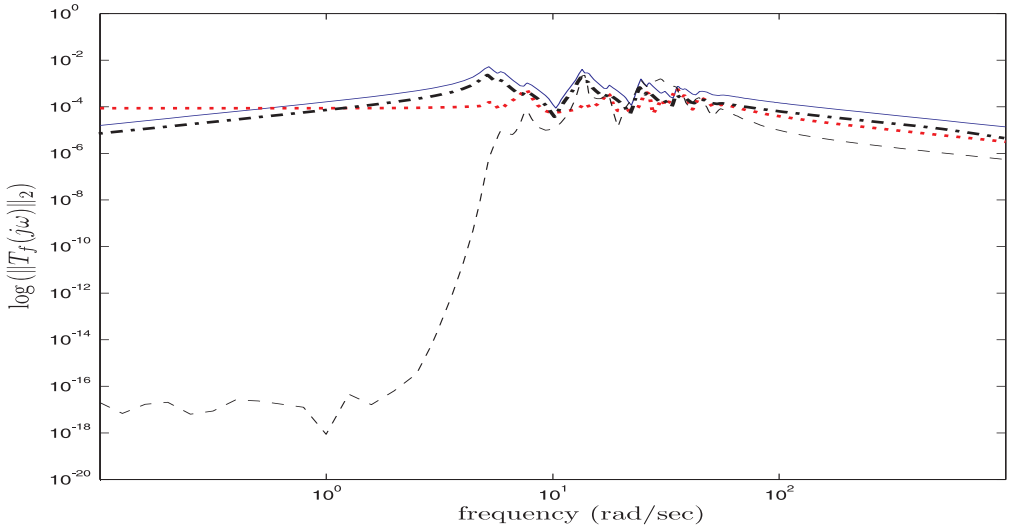


Fig. 5.7. $\sigma_{\max}$-plot of the frequency responses for Building model.
— full model, ··· BT error system,
-·- MLRS error system, - - RLRG error system.

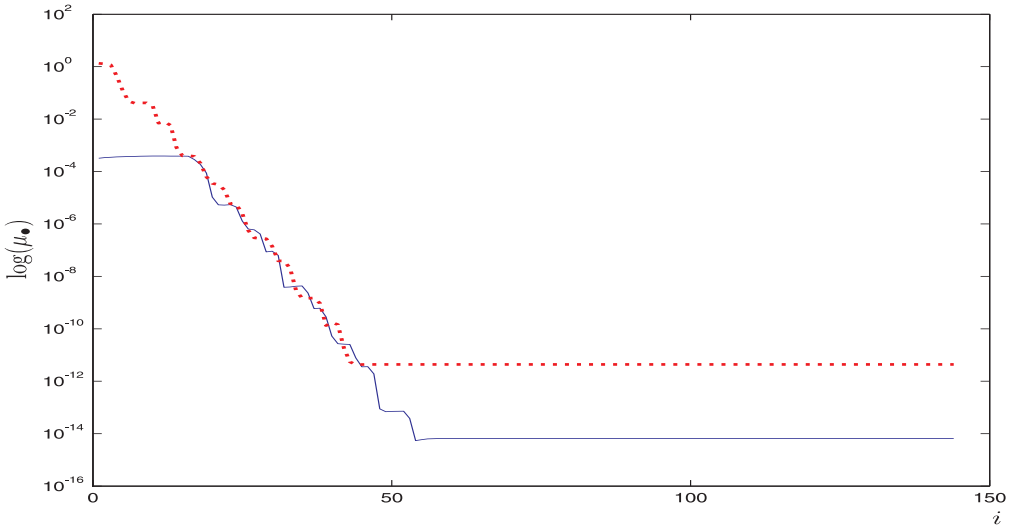


Fig. 5.8. Evolution of the values of $\mu_\bullet$ for Building model.
— μ_c and ··· μ_o.

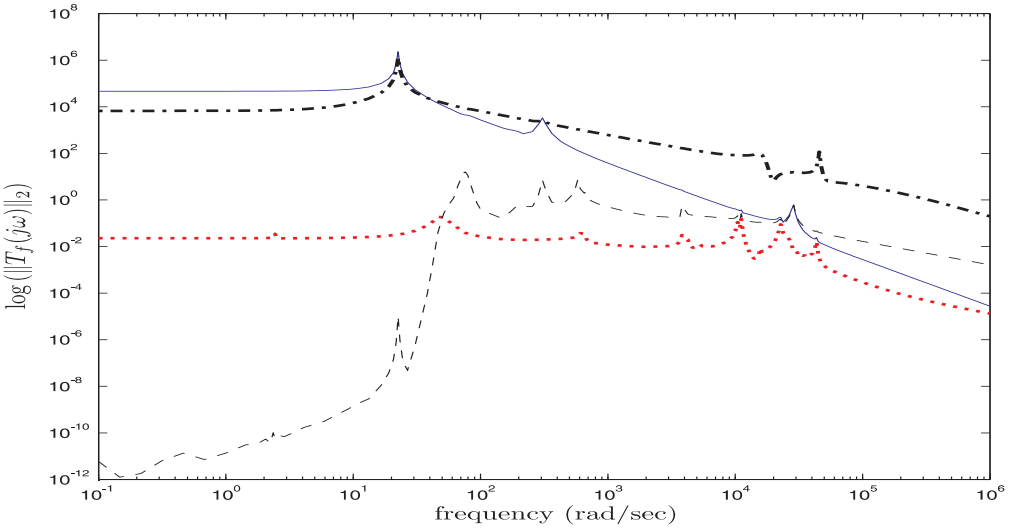


Fig. 5.9. $\sigma_{\max}$-plot of the frequency responses for CDplayer model.
— full model, ··· BT error system, -·- MLRS error system, -- RLRG error system.

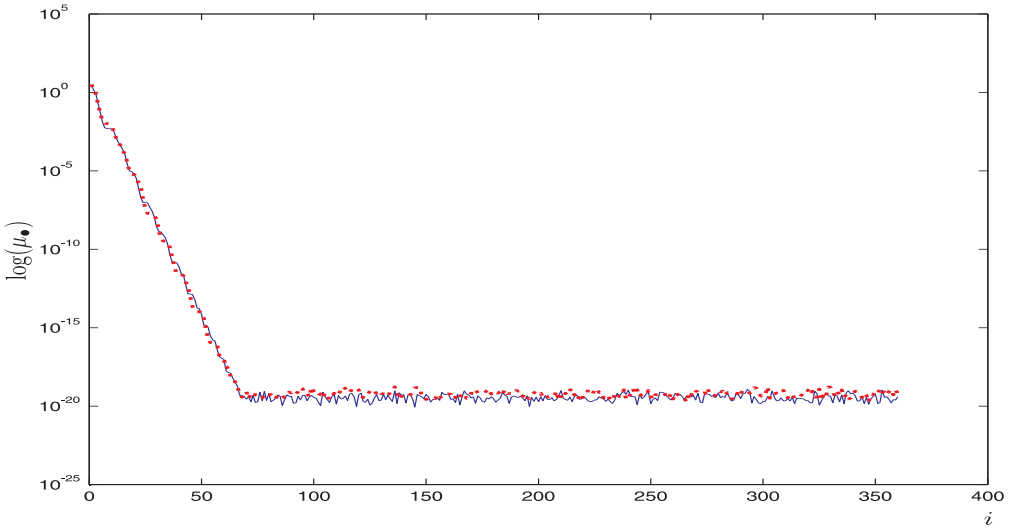


Fig. 5.10. Evolution of the values of $\mu_\bullet$ for CDplayer model.
— μ_c and ··· μ_o.

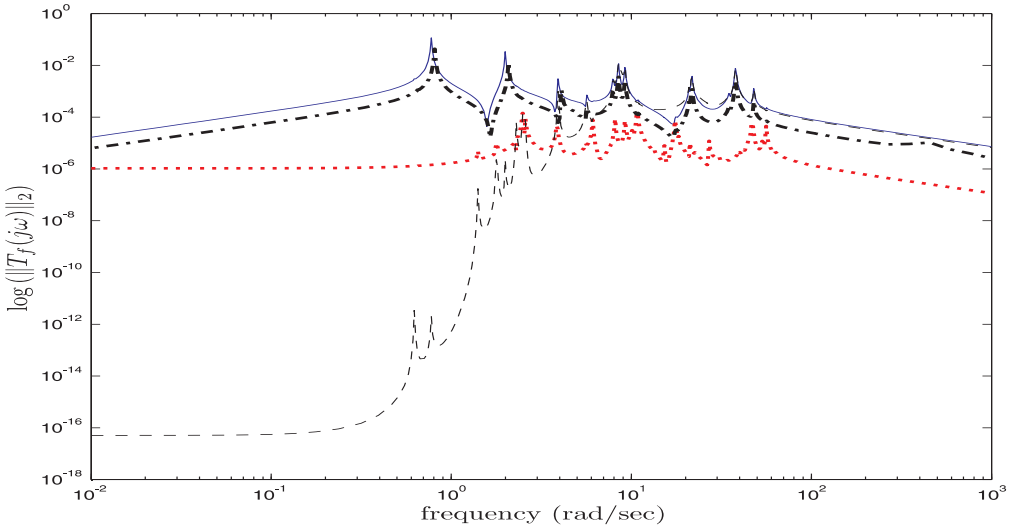


Fig. 5.11. $\sigma_{\max}$-plot of the frequency responses for ISS 1R model.
— full model, ··· BT error system,
-·- MLRS error system, - - RLRG error system.

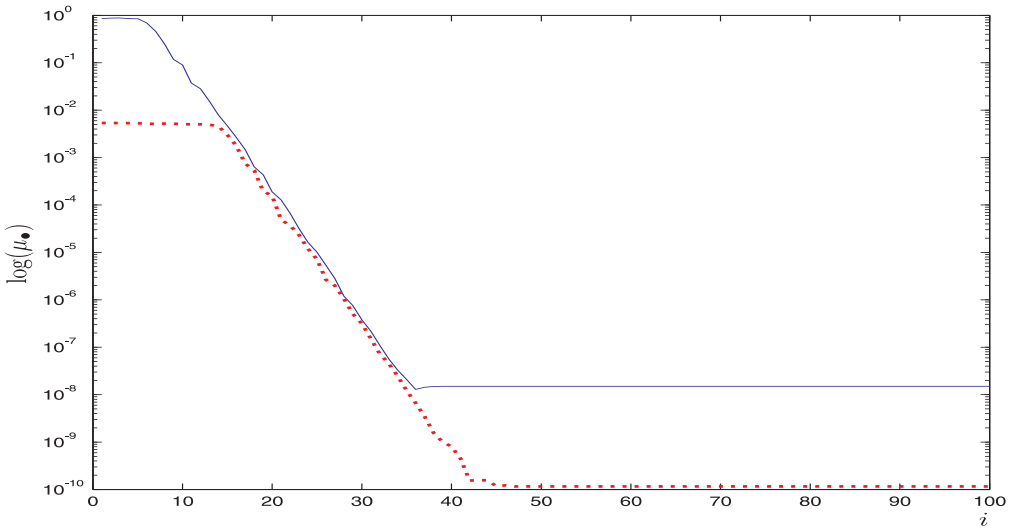


Fig. 5.12. Evolution of the values of $\mu_\bullet$ for ISS 1R model.
— μ_c and ··· μ_o.

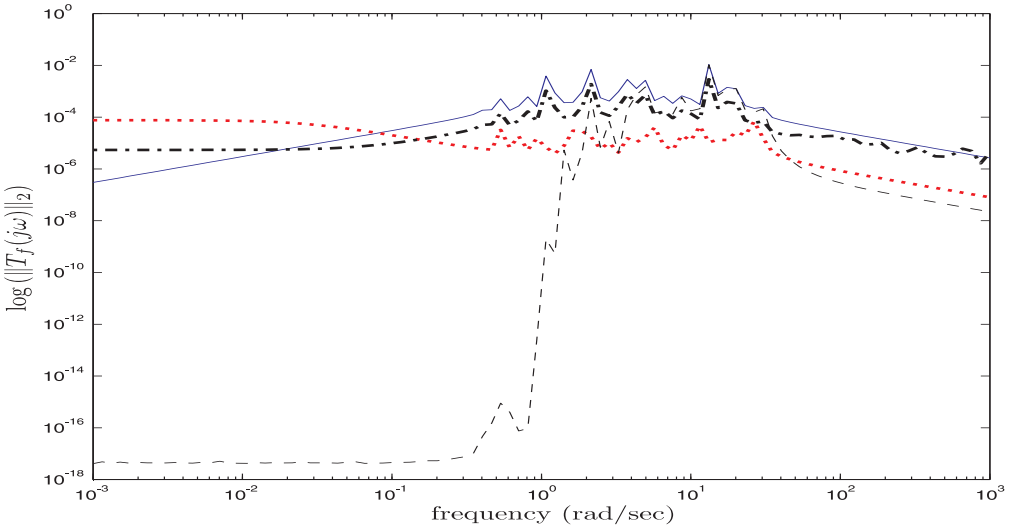


Fig. 5.13. $\sigma_{\max}$-plot of the frequency responses for ISS 12A model.
— full model, ··· BT error system,
-·- MLRS error system, - - RLRG error system.

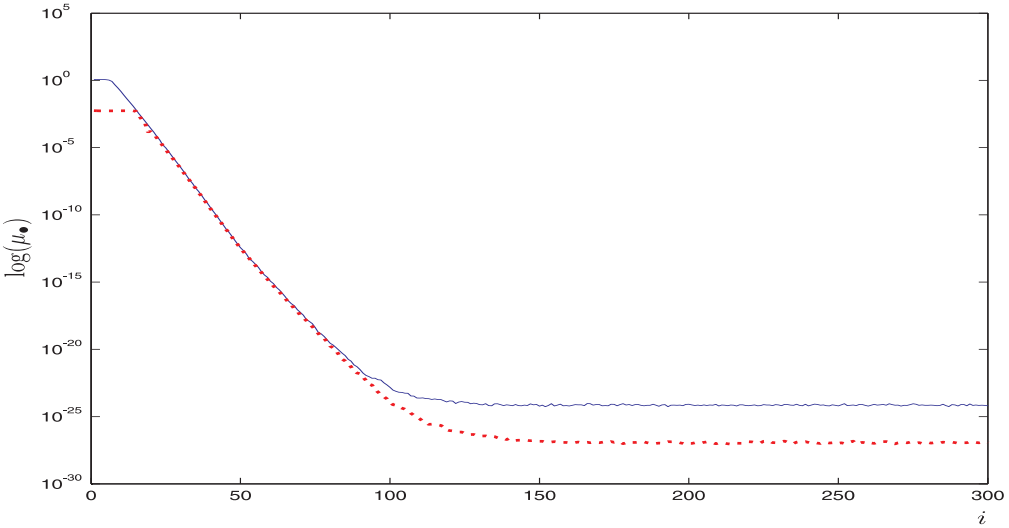


Fig. 5.14. Evolution of the values of $\mu_\bullet$ for ISS 12A model.
— μ_c and ··· μ_o.

5.6 Concluding remarks

In the last two chapters we proposed two recursive model reduction methods based on Gramians. The first algorithm is the MLRS algorithm for which we propose a detailed numerical study, with applications to image reconstruction and model reduction. The second algorithm is a new algorithm, namely the RLRG algorithm. Unlike the MLRS algorithm, RLRG is mainly aimed at the time-varying computation of approximate Gramians. This approach provides results that are close to those obtained by Balanced truncation with a better computational performance.
The RLRG procedure operates as follows. Given an initial estimate of this dominant subspace, it applies A to the old space and then performs a correction using B. This procedure converges to the dominant subspace of the square root of the Gramian. The principal advantage of this method is that it decreases the effect of the

old approximation errors since they are weighted at each iteration by a coefficient lower than 1 (the spectral radius of A which is strictly smaller than 1 if we assume that the system is stable), and thus only the last errors dominate. Those can be made very small by adapting (in a dynamic way) the initial starting point. Indeed, one can run the algorithm with any initial starting point, and as the noise levels become smaller we restart the algorithm with as initial starting point the low-rank approximation obtained. This rerun of the algorithm gives better results for the quality of the low-rank approximations and the smallness of the noise levels. And so the noise level (i.e., μ_c and μ_o) resulting from the RLRG algorithm are in fact dominated by the last terms which give much tighter bounds on the quality of the approximation. Moreover, as the noise levels resulting from the RLRG algorithm are small, the approximation of the product of the Gramians is better than the one obtained with the MLRS algorithm.

We illustrated the performance of this procedure using benchmark examples. These numerical examples show that this method is much better than the MLRS procedure and yields results closer to those obtained using Balanced Truncation.

Despite the obviously desirable features of the Gramians approach proposed here, many open questions remain. There are a number of questions about stability and properties of the original system retained in the reduced-order system. We also still need a complete description of the fixed points and their properties.

Chapter 6

Recursive Low-Rank Hankel approximation (RLRH)

Our first two algorithms have a common idea which is the independent approximation of the two Gramians. So to obtain a reduced model we have to "balance" the projection matrices obtained from these two approximations. The quality of the approximation and indeed of the reduced model depends on two parameters μ_c and μ_o which determine if the two Gramians are well approximated or not. These parameters are independent as we approximate Gramians independently from one another, and so one can imagine the case where one Gramian is well approximated and the other not. So, this affects the quality of the approximation of the reduced model.

In this chapter we present an algorithm which avoids this kind of problem. The key idea of this approach is to use the underlying recurrences defining the so-called time-varying "Hankel maps". These matrices have a similar structure to the Hankel matrix of the time-invariant case. Because the system order at each instant is given by the rank of the Hankel matrix at that instant, one can approximate the system by approximating the Hankel matrix.
Actually, this is the idea of the exact Hankel norm approximation methods [48]. In this case, the norm approximation problem is

$$\min_{rank\hat{\mathcal{H}}\leq n} \|\mathcal{H}-\hat{\mathcal{H}}\|, \tag{6.1}$$

where $\mathcal{H}$ is the Hankel map which makes correspondence between inputs and outputs. The problem (6.1) has many solutions, since only the largest singular values of the difference $E=\mathcal{H}-\hat{\mathcal{H}}$ is minimized, and $n-1$ others are free, as long as they remain smaller.
In general, to solve this problem, one has to select an appropriate representation of the desired high-order model that can be used computationally. A simple but high-complexity realization is given by the generalized companion form. Now, given this realization one can solve the problem (6.1) for a given precision which is measured using a Hermitian, strictly positive diagonal operator Γ (in fact it could be taken as $\Gamma=\epsilon.I$ for some small value of ϵ), by solving

$$\sup_k \| \left((\mathcal{H}-\hat{\mathcal{H}})\Gamma^{-1} \right)_k \| \leq 1$$

i.e., $\hat{\mathcal{H}}$ approximates $\mathcal{H}$ up to a precision given by Γ. This problem can be solved using the Schur-Takagi algorithm [48]. Indeed, Hankel norm approximation theory originates as a special case of the solution to the Schur-Takagi interpolation problem in the context of complex function theory. Several techniques were presented to find the optimal solution, see e.g. the work of Dewilde and van der Veen [48, 125], and Chandrasekaran and Gu [35, 36, 37, 38, 39]. The complexity of these techniques are normally of the order of $O(N^2)$ but can be made "fast" or "super fast" to be just of the order of $O(N)$. But in order to obtain this speed up, the matrices involved must have a special structure called the "*sequentially semi-separable matrix structure*". This structure involves some rank conditions for optimality and which cause some minor complications. This whole procedure has to be repeated for $\Gamma=c_k.I$, where c_k eventually converges to a small optimal value.

The principal idea of these algorithms is to use the SVD to approximate the Hankel matrices by matrices having a Hankel structure. Our algorithm follows the same line. It has the particularity that it approximates the Hankel matrices at each instant by a low rank approximations in a finite window. Let us now formulate this in more detail.

6.1 The RLRH algorithm

Let us consider a time window $[k_i, k_f] = [k-\tau, k+\tau-1]$ of width 2τ and centered around $[k-1, k]$. If we restrict the inputs to be non-zero only in the interval $[k_i, k-1]$ (i.e., the "past"), then the outputs in the interval $[k, k_f]$ (i.e., the "future") are given by the convolution with a "Hankel map". Indeed, the state-to-outputs and inputs-to-state maps on the finite window $[k_i, k_f]$ are given by $(k_i < k < k_f)$ [48] :

$$\underbrace{\begin{bmatrix} y_k \\ y_{k+1} \\ \vdots \\ y_{k_f} \end{bmatrix}}_{Y} = \underbrace{\begin{bmatrix} C_k \\ C_{k+1}A_k \\ \vdots \\ C_{k_f}\Phi(k_f,k) \end{bmatrix} \begin{bmatrix} B_{k-1} & A_{k-1}B_{k-2} & \dots & \Phi(k,k_i+1)B_{k_i} \end{bmatrix}}_{x(k)} \underbrace{\begin{bmatrix} u_{k-1} \\ u_{k-2} \\ \vdots \\ u_{k_i} \end{bmatrix}}_{U},$$

and the finite dimensional "Hankel" map $\mathcal{H}(k_f, k, k_i)$ mapping U to Y is :

$$\mathcal{H}(k_f,k,k_i) = \begin{bmatrix} C_kB_{k-1} & C_kA_{k-1}B_{k-2} & \dots & C_k\Phi(k,k_i+1)B_{k_i} \\ C_{k+1}A_kB_{k-1} & C_{k+1}A_kA_{k-1}B_{k-2} & & C_{k+1}\Phi(k+1,k_i+1)B_{k_i} \\ \vdots & & \ddots & \vdots \\ C_{k_f}\Phi(k_f,k)B_{k-1} & C_{k_f}\Phi(k_f,k-1)B_{k-2} & \dots & C_{k_f}\Phi(k_f,k_i+1)B_{k_i} \end{bmatrix}.$$

Since $x(k) \in \mathbb{R}^N$, $\mathcal{H}(k_f, k, k_i)$ is of rank at most N and it has a factorization :

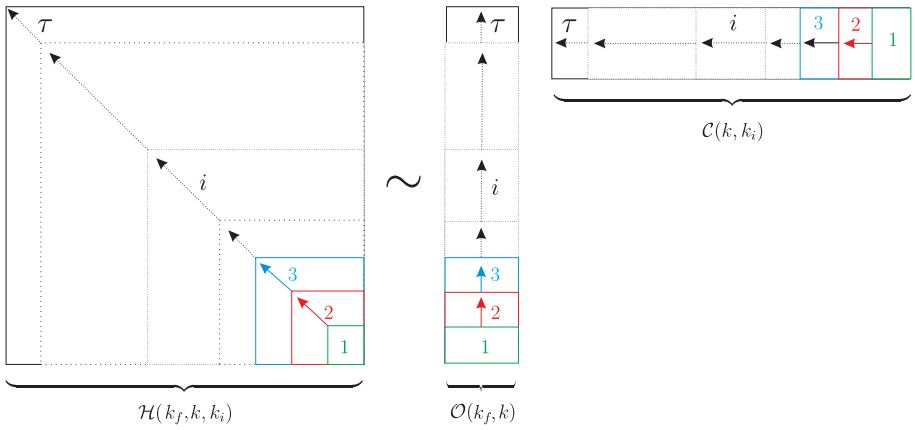
$$\mathcal{H}(k_f,k,k_i) = \underbrace{\begin{bmatrix} C_k \\ C_{k+1}A_k \\ \vdots \\ C_{k_f}\Phi(k_f,k) \end{bmatrix}}_{\mathcal{O}(k_f,k)} \underbrace{\begin{bmatrix} B_{k-1} & A_{k-1}B_{k-2} & \dots & \Phi(k,k_i+1)B_{k_i} \end{bmatrix}}_{\mathcal{C}(k,k_i)},$$

where $\mathcal{O}_k \doteq \mathcal{O}(k_f, k)$ and $\mathcal{C}_k \doteq \mathcal{C}(k, k_i)$ are respectively the observability and the controllability matrices at instant k related respectively to the finite windows $[k, k_f]$ and $[k_i, k-1]$. It also follows from the factorization that the submatrices of the factors satisfy the following recurrences :

$$\mathcal{O}_j = \begin{bmatrix} C_j \\ \mathcal{O}_{j+1}A_j \end{bmatrix}, \quad j = k_f-1, \dots, k, \quad \mathcal{O}_{k_f} = C_{k_f},$$

$$\mathcal{C}_{j+1} = \begin{bmatrix} B_j & A_j\mathcal{C}_j \end{bmatrix}, \quad j = k_i+1, \dots, k-1, \quad \mathcal{C}_{k_i+1} = B_{k_i}.$$

These recurrences construct the controllability matrix forward from k_i to k and the observability matrix backward from k_f to k. The idea of our Recursive Low-Rank Hankel approximation method is now to compute these recurrences using low-rank approximations at each time step, according to the following recursive scheme :



Algorithm 7 The Recursive Low-Rank Hankel algorithm (RLRH).

- Let the initializing matrices $S_n(k_i), R_n(k_f) \in \mathbb{R}^{N\times n}$ are

$$S_n(k_i) = \mathbf{0}, \quad R_n(k_f) = \mathbf{0}.$$

- Then the i^{th} $(i = 1,\dots,\tau)$ low-rank approximations $S_n(k_i+i)$ and $R_n(k_f-i)$ are obtained as follows.

$$\left[\, S_n(k_i+i) \middle| E_c(k_i+i) \,\right] = \left[\, B_{k_i+i-1} \middle| A_{k_i+i-1} S_n(k_i+i-1) \,\right] \left[\, V^{(1)}(i) \middle| V^{(2)}(i) \,\right] \tag{6.2}$$

$$\left[\begin{array}{c} R_n^T(k_f-i) \\ \hline E_o^T(k_f-i) \end{array}\right] = \left[\begin{array}{c} U^{(1)}(i)^T \\ \hline U^{(2)}(i)^T \end{array}\right] \left[\begin{array}{c} C_{k_f-i} \\ \hline R_n^T(k_f-i+1)A_{k_f-i} \end{array}\right], \tag{6.3}$$

where $U^{(1)}(i) \in \mathbb{R}^{(p+n)\times n}$ and $V^{(1)}(i) \in \mathbb{R}^{(m+n)\times n}$ come from the SVD of the product :

$$\left[\begin{array}{c} C_{k_f-i} \\ \hline R_n^T(k_f-i+1)A_{k_f-i} \end{array}\right] \cdot \left[\, B_{k_i+i-1} \middle| A_{k_i+i-1} S_n(k_i+i-1) \,\right] = \underbrace{U(i)\Sigma(i)V(i)^T}_{\text{SVD}}. \tag{6.4}$$

Remark 6.1.

- For the initialization of the algorithm we have choose

$$S_n(k_i) = \mathbf{0}, \quad \text{and} \quad R_n(k_f) = \mathbf{0},$$

which imply that

$$E_c(k_i) = \mathcal{C}(k_i,k_i), \quad \text{and} \quad E_o(k_f) = \mathcal{O}^T(k_f,k_f);$$

- It follows from (6.4) that at each iteration $i = 1,\dots,\tau$ we have

$$\left[\begin{array}{c} R_n^T(k_f-i) \\ \hline E_o^T(k_f-i) \end{array}\right] \left[\, S_n(k_i+i) \middle| E_c(k_i+i) \,\right] = \left[\begin{array}{c|c} \Sigma_1(i) & 0 \\ \hline 0 & \Sigma_2(i) \end{array}\right] \tag{6.5}$$

- Let us investigate the amount of work involved in our algorithm : first we need to form products of the type $A_j S_n(j)$ and $R_n^T(l+1)A_l$. If we assume the matrices A_i to be sparse[1], then the amount of work needed for this is $O(\alpha N n)$ [63]. The construction of the left hand side of (6.4) requires an additional $2N(n+m)(n+p)$ flops and the application of the transformations U and V requires $O((p+n)(m+n)(2n+p+m))$ flops, and so the complexity of this algorithm is $O(N(p+n)(m+n))$ for each iteration.

[1] α the number of non-zero elements per row or column of A_i.

Now in order to make the link between the whole controllability and observability matrices $\mathcal{C}(\cdot,\cdot)$, $\mathcal{O}(\cdot,\cdot)$ and their low-rank approximations $S_n(\cdot)$ and $R_n(\cdot)$, we have the following theorem :

Theorem 6.2.
At each iteration, there exist orthogonal matrices $V_i \in \mathbb{R}^{(n+im)\times(n+im)}$ and $U_i \in \mathbb{R}^{(n+ip)\times(n+ip)}$ satisfying :

$$\mathcal{C}(k_i+i,k_i)V_i = \left[\, S_n(k_i+i) \middle| E_c(k_i+i) \middle| A_{k_i+i-1}C_e(k_i+i,k_i) \,\right]$$

$$\mathcal{O}^T(k_f,k_f-i)U_i = \left[\, R_n(k_f-i) \middle| E_o(k_f-i) \middle| A^T_{k_f-i}O_e(k_f-i+1,k_f) \,\right]$$

where $E_c(k_i+i)$ and $E_o(k_f-i)$ are the neglected parts at iteration i in (6.2) and (6.3), and the matrices $\mathcal{C}_e(j,k_i)$ and $\mathcal{O}_e(k_f,j)$ are defined as follows :

$$\mathcal{C}_e(j,k_i) \doteq \left[\, E_c(j-1) \middle| \ldots \middle| \Phi(j-1,k_i)E_c(k_i) \,\right],$$

and

$$\mathcal{O}_e(k_f,j)^T \doteq \left[\, E_o(j) \middle| \ldots \middle| \Phi(k_f,j)^T E_o(k_f) \,\right].$$

■

Proof.
We just show the proof for V_i, the other is similar.
At each step, there exists an orthogonal matrix $V(i) = \left[\, V_1(i) \middle| V_2(i) \,\right]$ such that

$$\left[\, B_i \middle| A_iS_n(i) \,\right] V(i) = \left[\, S_n(i+1) \middle| E_c(i+1) \,\right].$$

For $i = k_i$ we have $\mathcal{C}(k_i,k_i) = \left[\, S_n(k_i) \middle| E_c(k_i) \,\right]$, and so $V_0 = I$.
We prove the general result by induction. Suppose that there exists an orthogonal matrix V_i such that

$$\mathcal{C}(i,k_i)V_i = \left[\, S_n(i) \middle| E_c(i) \middle| A_{i-1}E_c(i-1) \middle| \ldots \middle| \Phi(i,k_i)E_c(k_i) \,\right].$$

Since $\mathcal{C}(i+1,k_i) = \left[\, B_i \middle| A_i\mathcal{C}(i,k_i) \,\right]$, we choose

$$V_{i+1} = \begin{bmatrix} I_m & 0 \\ 0 & V_i \end{bmatrix} \begin{bmatrix} V(i) & 0 \\ 0 & I_{(i+1)m} \end{bmatrix},$$

from which it follows that

$$\begin{aligned}
\mathcal{C}(i+1,k_i)V_{i+1} &= \left[\, B_i \middle| A_i\mathcal{C}(i,k_i) \,\right] \begin{bmatrix} I_m & 0 \\ 0 & V_i \end{bmatrix} \begin{bmatrix} V(i) & 0 \\ 0 & I_{(i+1)m} \end{bmatrix} \\
&= \left[\, B_i \middle| A_i\mathcal{C}(i,k_i)V_i \,\right] \begin{bmatrix} V(i) & 0 \\ 0 & I_{(i+1)m} \end{bmatrix} \\
&= \left[\, B_i \middle| A_iS_n(i) \middle| A_iE_c(i) \middle| \ldots \middle| \Phi(i+1,k_i)E_c(k_i) \,\right] \begin{bmatrix} V(i) & 0 \\ 0 & I_{(i+1)m} \end{bmatrix} \\
&= \left[\, S_n(i+1) \middle| E_c(i+1) \middle| A_iE_c(i) \middle| \ldots \middle| \Phi(i+1,k_i)E_c(k_i) \,\right] \\
&= \left[\, S_n(k_i+i) \middle| E_c(k_i+i) \middle| A_{k_i+i-1}C_e(k_i+i,k_i) \,\right].
\end{aligned}$$

□

As a consequence of this theorem we have the following result which give us an approximation of the original Hankel map $\mathcal{H}(k_f,k,k_i)$:

Theorem 6.3.

At each instant k, there exist orthogonal matrices $V_\tau \in \mathbb{R}^{(n+\tau m)\times(n+\tau m)}$ and $U_\tau \in \mathbb{R}^{(n+\tau p)\times(n+\tau p)}$ such that :

$$U_\tau^T \mathcal{H}(k_f, \tau, k_i) V_\tau = \tag{6.6}$$

$$\left[\begin{array}{c|c|c} R_n^T(\tau) S_n(\tau) & \boldsymbol{0} & R_n^T(\tau) A_{\tau-1} \mathcal{C}_e(\tau, k_i) \\ \hline \boldsymbol{0} & E_o^T(\tau) E_c(\tau) & E_o^T(\tau) A_{\tau-1} \mathcal{C}_e(\tau, k_i) \\ \hline \mathcal{O}_e(k_f, \tau{+}1) A_\tau S_n(\tau) & \mathcal{O}_e(k_f, \tau{+}1) A_\tau E_c(\tau) & \mathcal{O}_e(k_f, \tau{+}1) A_\tau A_{\tau-1} \mathcal{C}_e(k, k_i) \end{array}\right].$$

■

Proof. First we have the relationship between the Hankel operators, the controllability and the observability matrices :

$$\mathcal{H}(k_f, k, k_i) \doteq \mathcal{O}(k_f, k) \mathcal{C}(k, k_i)$$

and from the theorem (6.2), there exist two orthogonal matrices $V_\tau \in \mathbb{R}^{(n+\tau m)\times(n+\tau m)}$ and $U_\tau \in \mathbb{R}^{(n+\tau p)\times(n+\tau p)}$ such that :

$$\begin{aligned} U_\tau^T \mathcal{H}(k_f, \tau, k_i) V_\tau &\doteq U_\tau^T \mathcal{O}(k_f, \tau) \mathcal{C}(\tau, k_i) V_\tau \\ &= \left[\begin{array}{c} R_n^T(\tau) \\ \hline E_o^T(\tau) \\ \hline O_e(\tau{+}1, k_f) A_\tau \end{array}\right] \left[\, S_n(\tau) \,\middle|\, E_c(\tau) \,\middle|\, A_{\tau-1} \mathcal{C}_e(\tau, k_i) \,\right] \end{aligned}$$

The final result follows from the remark (6.5). □

This result enables us to evaluate the quality of our approximations by using the Hankel operator without having to pass by Gramians, which can be very suitable in some cases.

6.2 The time-invariant case

Let us analyze the quality of our approximation for the time-invariant case. In this special case, we run the RLRH algorithm on the interval $[-\tau, \tau]$ which means that we choose $k_i = -\tau$ and $k_f = +\tau$ ($\tau \in \mathbb{N}$).
The procedure yields two matrices S_n and R_n of full rank n. Using those matrices we can approximate the Gramians $\mathcal{G}_c$ and $\mathcal{G}_o$ of the original model $\mathcal{S}$ by $S_n S_n^T$ and $R_n R_n^T$, respectively. The differences between the approximate low-rank Gramians and the exact Gramians

$$\mathcal{E}_c(k) \doteq \mathcal{G}_c(k) - \mathcal{P}_k, \quad \mathcal{E}_o(k) \doteq \mathcal{G}_o(k) - \mathcal{Q}_k$$

remains bounded for large τ, as indicated in the following theorem.

Theorem 6.4.

Let $\mathcal{P}$ and $\mathcal{Q}$ be respectively the solutions of

$$\mathcal{P} = A \mathcal{P} A^T + I, \quad \textit{and} \quad \mathcal{Q} = A^T \mathcal{Q} A + I.$$

Then

$$\|\mathcal{E}_c(\tau)\|_2 \leq \eta_c^2 \|\mathcal{P}\|_2 \leq \eta_c^2 \frac{\kappa(A)^2}{1 - \rho(A)^2},$$

and

$$\mathcal{E}_o(\tau)\|_2 \leq \eta_o^2 \|\mathcal{Q}\|_2 \leq \eta_o^2 \frac{\kappa(A)^2}{1 - \rho(A)^2}$$

where $\eta_c \doteq \max_k \|E_c(k)\|_2$ *and* $\eta_o \doteq \max_k \|E_o(k)\|_2$. ■

Proof.

It follows from Theorem 6.2 that

$$\mathcal{E}_c(i+1) = A\mathcal{E}_c(i)A^T + E_c(i)E_c(i)^T, \quad \text{and} \quad \mathcal{E}_o(i+1) = A^T\mathcal{E}_o(i)A + E_o(i)E_o(i)^T.$$

We can also consider the equations :

$$\mathcal{X}_c(i+1) = A\mathcal{X}_c(i)A^T + (\eta_c^2 I - E_c(i)E_c(i)^T), \quad \mathcal{X}_c(0) = 0,$$

$$\mathcal{X}_o(i+1) = A^T\mathcal{X}_o(i)A + (\eta_o^2 I - E_o(i)E_o(i)^T), \quad \mathcal{X}_o(0) = 0.$$

Their iterates $\mathcal{X}_c(i)$ and $\mathcal{X}_o(i)$ are clearly positive semi-definite and hence converge to a solution $\mathcal{X}_c$ and $\mathcal{X}_o$ which are also positive semi-definite.
Moreover by linearity we have

$$\mathcal{E}_c(i+1) + \mathcal{X}_c(i+1) = A(\mathcal{E}_c(i) + \mathcal{X}_c(i))A^T + \eta_c^2 I,$$

$$\mathcal{E}_o(i+1) + \mathcal{X}_o(i+1) = A^T(\mathcal{E}_o(i) + \mathcal{X}_o(i))A + \eta_o^2 I,$$

It then follows that

$$\lim_{i\to\infty} \mathcal{E}_c(i) + \mathcal{X}_c(i) = \eta_c^2\mathcal{P}, \quad \text{and} \quad \lim_{i\to\infty} \mathcal{E}_o(i) + \mathcal{X}_o(i) = \eta_o^2\mathcal{Q}$$

and we obtain

$$\|\mathcal{E}_c(i)\|_2 \le \eta_c^2\|\mathcal{P}\|_2, \quad \text{and} \quad \|\mathcal{E}_o(i)\|_2 \le \eta_o^2\|\mathcal{Q}\|_2.$$

The second bound follows from the eigen-decomposition of A. □

Theorem 6.5.

Using the first n columns $U_k^{(1)}$ of U_k and $V_k^{(1)}$ of V_k, we obtain a rank n approximation of the Hankel map :

$$\mathcal{H}(k) - U_k^{(1)}R_n^T(k)\cdot S_n(k)V_k^{(1)T} = \mathcal{E}_h(k),$$

for which we have the error bound :

$$\|\mathcal{E}_h(k)\|_2 \le \frac{\kappa(A)}{\sqrt{1-\rho(A)^2}}\max\{\eta_c\|R_n^T A\|_2, \eta_o\|AS_n\|_2\} + \frac{\kappa(A)^2}{1-\rho(A)^2}\eta_o\eta_c.$$

■

Proof.

This follows directly from the bounds of Theorem (6.3) that can be used to bound the blocks in the form in (6.6) different from the $(1,1)$ block.
More explicitly, from (6.6) we have

$$\mathcal{H}(k_f,k,k_i) = U_k \left[\begin{array}{c|cc} R_n^T(k)S_n(k) & \mathbf{0} & R_n^T(k)A\mathcal{C}_e(k,k_i) \\ \hline \mathbf{0} & E_o^T(k)E_c(k) & E_o^T(k)A\mathcal{C}_e(k,k_i) \\ \mathcal{O}_e(k_f,k+1)AS_n(k) & \mathcal{O}_e(k_f,k+1)AE_c(k) & \mathcal{O}_e(k_f,k+1)A^2\mathcal{C}_e(k,k_i) \end{array}\right] V_k^T$$

and so

$$\mathcal{E}_h(k) = \mathcal{E}_h^{(1)}(k) + \mathcal{E}_h^{(2)}(k)$$

where

$$\mathcal{E}_h^{(1)}(k) = U_k \left[\begin{array}{c|cc} \mathbf{0} & \mathbf{0} & R_n^T(k)A\mathcal{C}_e(k,k_i) \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathcal{O}_e(k_f,k+1)AS_n(k) & \mathbf{0} & \mathbf{0} \end{array}\right] V_k^T$$

and

$$\mathcal{E}_h^{(2)}(k) = U_k \left[\begin{array}{c|cc} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & E_o^T(k)E_c(k) & E_o^T(k)A\mathcal{C}_e(k,k_i) \\ \mathbf{0} & \mathcal{O}_e(k_f,k+1)AE_c(k) & \mathcal{O}_e(k_f,k+1)A^2\mathcal{C}_e(k,k_i) \end{array}\right] V_k^T$$

and thus

$$\begin{aligned} \|\mathcal{E}_h(k)\|_2 \leq \max\{&\|R_n^T(k)A\mathcal{C}_e(k,k_i)\|_2, \|\mathcal{O}_e(k_f,k+1)AS_n(k)\|_2\} + \\ &\left\| \begin{bmatrix} E_o^T(k)E_c(k) & E_o^T(k)A\mathcal{C}_e(k,k_i) \\ \mathcal{O}_e(k_f,k+1)AE_c(k) & \mathcal{O}_e(k_f,k+1)A^2\mathcal{C}_e(k,k_i) \end{bmatrix} \right\|_2 \end{aligned}$$

□

Remark 6.6.

In the time-invariant case all matrices A, B and C are constant. As a consequence all Hankel maps are equal as well and only the interval width plays a role in the obtained decomposition. As a consequence one obtains an approximate rank factorization of a Hankel map with i block columns and rows at each instant i. The bounds obtained in Theorem 6.4 and 6.5 are moreover independent of k. As i grows larger one can expect that a reasonable approximation of η_c and η_o are in fact given by the last terms, i.e., $\eta_c \approx \|E_c(k)\|_2$ and $\eta_o \approx \|E_c(k)\|_2$ which will give much tighter bounds in these theorems. In fact, as in the remark page 5.5, η_c and η_o are function of k_0 and one can write

$$\eta_c(-\tau) = \max_{-\tau \leq i \leq \infty} \|E_c(i)\|_2, \quad \text{and} \quad \eta_o(\tau) = \max_{-\infty \leq i \leq \tau} \|E_o(i)\|_2.$$

Since $\eta_c(k_0)$ and $\eta_o(k_0)$ are typically decreasing we can replace it by the maximum over the last iteration steps.

6.3 RLRG versus RLRH

Now, let us compare the two algorithms (RLRG and RLRH) for model reduction of large scale systems. First, investigating the amount of work involved by both algorithms give :

- For both methods one needs to form products of the type $A_j S_n(j)$ and $R_n^T(l+1)A_l$. If we assume the matrices A_k to be sparse, then the amount of work needed for this is $O(\alpha N n)$ where α is the number of non-zero elements per row or column of A_k [63];
- For the RLRG method, we need to compute and apply at each step the transformation U_c, U_o. This requires $O(N(n+m)^2)$ flops and $O(N(n+p)^2)$ flops, respectively [63];
- For the RLRH method, the construction of the left hand side of (6.4) requires an additional $2N(n+m)(n+p)$ flops and the application of the transformations U_h and V_h requires $O((p+n)(m+n)(2n+p+m))$ flops.

The two methods have thus a comparable complexity $O(N(n+m)(n+p))$ when the matrices A_k are sparse. But as RLRH works on the Hankel map rather than on the individual Gramians, it should suffer less from a bad balancing of the original system. This is illustrated in the examples of the next section.
Here also Theorem 5.4 is still available and we obtain a much tighter bound than those obtained for the Modified Low-Rank Smith and the Recursive Low-Rank Gramian algorithms.

Notice that since we are defining projectors for finite time windows, these algorithms could be applied to linear time-invariant systems that are unstable. One can then not show any property of stability for the reduced order model, but the finite horizon Hankel map will at least be well approximated.

6.4 Numerical examples

In this section we compare the numerical results of the RLRH algorithm and the numerical results of the previous algorithms applied to benchmark examples.
For this case, the reduced-order model is also obtained via an approximated balanced truncation. But remark that because we work directly on the Hankel map we do not need to "balance" (using an SVD) the projection matrices to obtain a convenient reduced-order model.
For each example, the relative $\mathcal{H}_\infty$ norms of the full system and the error systems are tabulated, and the σ_{max}-plot of the full order and the corresponding error system are shown. It can be seen from Figures 6.1, 6.3, 6.5, and 6.7 that we obtain, with RLRH approximation, results which are close to those obtained via BT. These results are also close of those of RLRG approximation, but we have applied the RLRG algorithm to the controllability and observability matrices with a $\hat{n}$, where $\hat{n} > n$, and we have "balanced" the projection matrices using a SVD to keep only n projection matrices. These operations make the RLRG more expensive, and so the RLRH algorithm is less expensive and the results are as good as those obtained using the RLRG approximation.
Figures 6.2, 6.4, 6.6, and 6.8 shows the noise levels. Notice that these noise levels shown must be interpreted also in a special way as it was done for the RLRG algorithm. The noise levels must be multiplied by the corresponding power of the spectral radius of A to obtain the real values of the noise level at the end, i.e., the real noise level $\tilde{\mu}_\bullet$ is obtained as

$$\tilde{\mu}_\bullet(i) \doteq \rho(A)^{\tau-i}\mu_\bullet(i), \quad \text{where } \tau \text{ is the number of iteration.}$$

And so the values of noise levels considered in Theorems 6.3, 6.4, and 6.5 will be taken in the last obtained values which will be very small.
We notice here also that for poorly balanced systems the resulting noise levels are not of the same order as for well balanced systems. This is still the case for the CDplayer model.
We remark also that for "close balanced" systems, like the CDplayer model ($\kappa(T) = 40.7341$ where T is the balancing transformation) RLRG yields better results. But, RLRH is as least better for "poorly balanced" systems. This is the case for the Building model ($\kappa(T) = 347.0781$) and more clearly for the International space station ($\kappa(T) = 7.4018.10^5$). Of course, RLRH is always better in terms of cost.

model	$\|\mathcal{S}\|_\infty$	$\frac{\|\mathcal{S}-\mathcal{S}_n^{BT}\|_\infty}{\|\mathcal{S}\|_\infty}$	$\frac{\|\mathcal{S}-\mathcal{S}_n^{MLRS}\|_\infty}{\|\mathcal{S}\|_\infty}$	$\frac{\|\mathcal{S}-\mathcal{S}_n^{RLRG}\|_\infty}{\|\mathcal{S}\|_\infty}$	$\frac{\|\mathcal{S}-\mathcal{S}_n^{RLRH}\|_\infty}{\|\mathcal{S}\|_\infty}$
Building	0.0053	0.1143	0.9994	0.4301	0.4320
CDplayer	$2.3198.10^6$	$8.0704.10^{-8}$	0.9875	$6.8931.10^{-6}$	$1.7.10^{-6}$
ISS 1R	0.1159	0.0013	1.0000	0.1023	0.0979
ISS 12A	0.0107	0.0071	0.9992	0.9697	0.9390

Table 6.1. H_∞ norm of benchmark models, and the error systems.

model	μ_c^{MLRS}	μ_o^{MLRS}	μ_c^{RLRG}	μ_o^{RLRG}	μ_c^{RLRH}	μ_o^{RLRH}
Building	$3.4411.10^{-4}$	0.6655	$6.5063.10^{-15}$	$4.3799.10^{-12}$	$7.7292.10^{-15}$	$3.2445.10^{-11}$
CDplayer	6.3739	6.3234	$4.0575.10^{-20}$	$6.0341.10^{-20}$	$1.6867.10^{-14}$	$2.8846.10^{-13}$
ISS 1R	0.5494	0.0030	$1.5063.10^{-8}$	$1.1641.10^{-10}$	$6.2550.10^{-8}$	$1.6270.10^{-9}$
ISS 12A	0.8248	0.0029	$7.1973.10^{-25}$	$1.1751.10^{-27}$	$7.5093.10^{-22}$	$4.7292.10^{-25}$

Table 6.2. Noise levels $\mu_\bullet$ for benchmark models.

model	BT	MLRS	RLRG	RLRH
Building	0.3750	0.1720	0.3380	0.0810
CDplayer	0.7970	1.8750	0.7340	0.7030
ISS 1R	11.6720	9.4530	4.7350	2.5470
ISS 12A	$1.1327.10^3$	$0.6794.10^3$	$0.1029.10^3$	$0.0282.10^3$

Table 6.3. CPU time for different algorithms

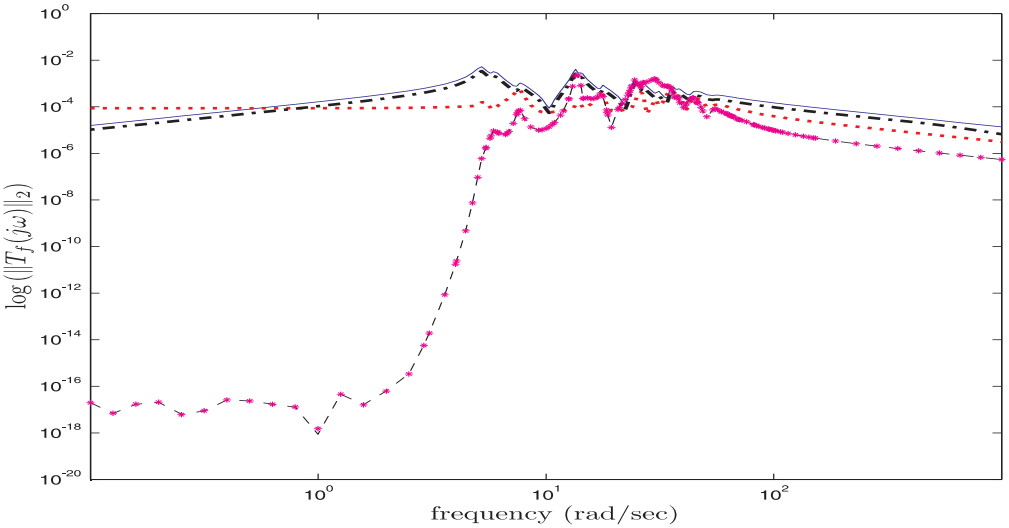


Fig. 6.1. $\sigma_{\max}$-plot of the frequency responses for Building model.
— full model, ··· BT error system, -·- MLRS error system,
-- RLRG error system, ∗ ∗ ∗ RLRH error system.

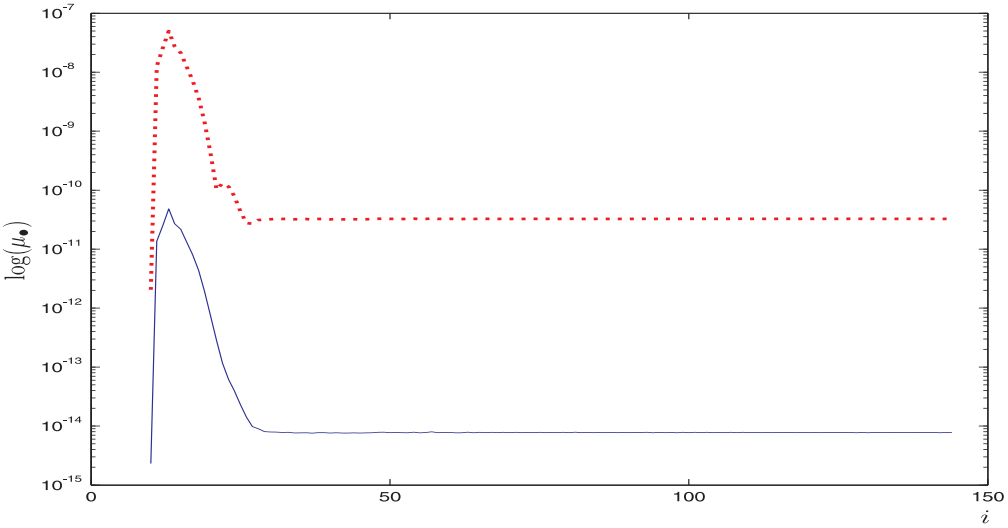


Fig. 6.2. Evolution of the values of $\mu_\bullet$ for Building model.
— μ_c and ··· μ_o.

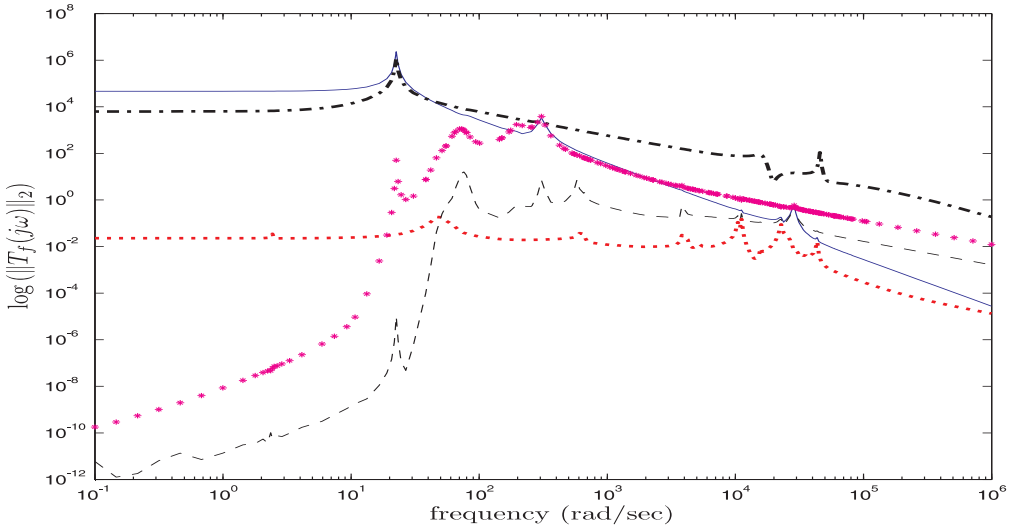


Fig. 6.3. $\sigma_{\max}$-plot of the frequency responses for CDplayer model.
— full model, ··· BT error system, -·- MLRS error system,
- - RLRG error system, ∗ ∗ ∗ RLRH error system.

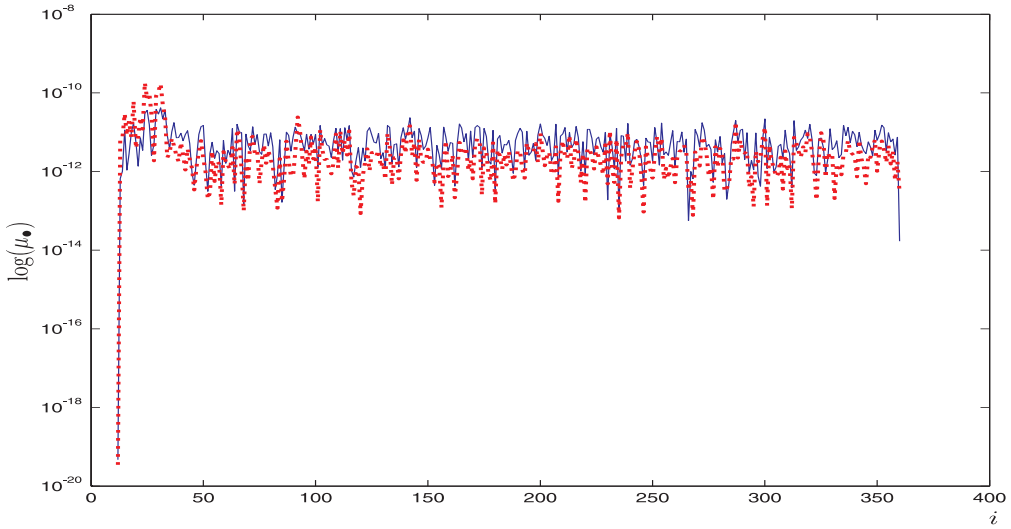


Fig. 6.4. Evolution of the values of $\mu_\bullet$ for CDplayer model.
— μ_c and ··· μ_o.

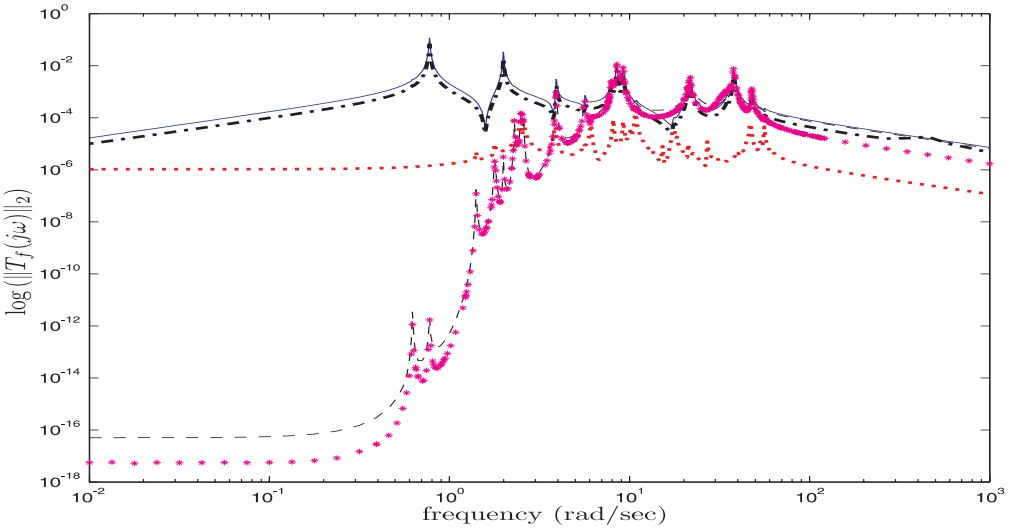


Fig. 6.5. $\sigma_{\max}$-plot of the frequency responses for ISS 1R model.
— full model, ··· BT error system, -·- MLRS error system,
-- RLRG error system, ∗ ∗ ∗ RLRH error system.

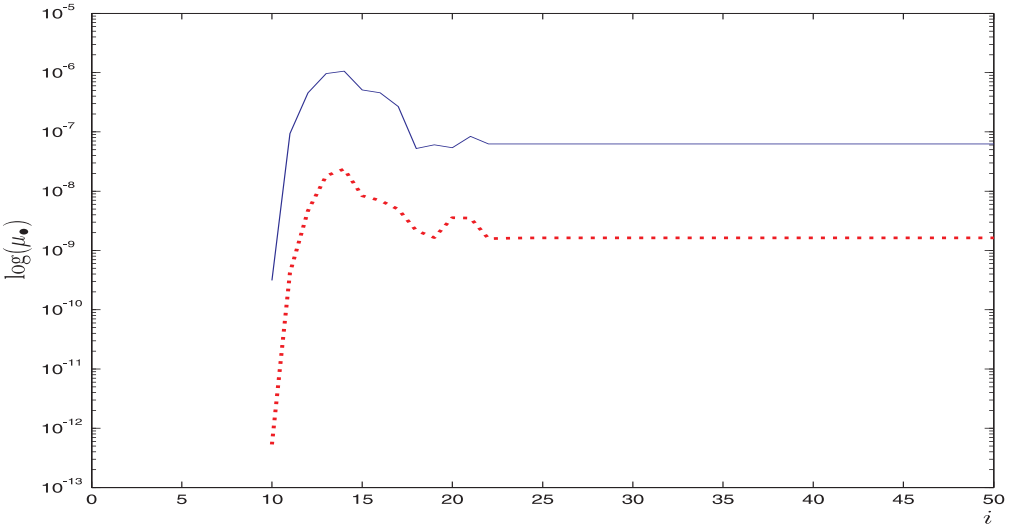


Fig. 6.6. Evolution of the values of $\mu_\bullet$ for ISS 1R model.
— μ_c and ··· μ_o.

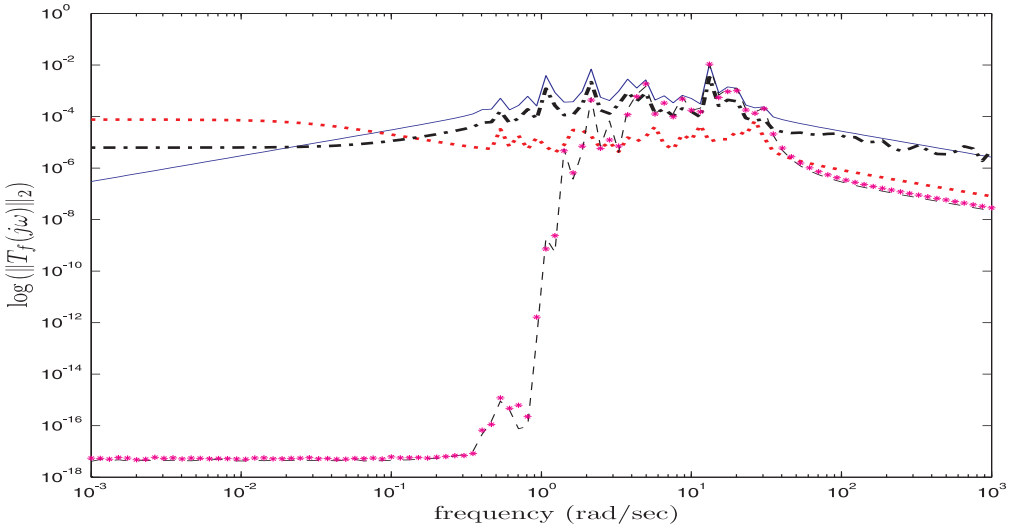


Fig. 6.7. $\sigma_{\max}$-plot of the frequency responses for ISS 12A model.
— full model, ··· BT error system, -·- MLRS error system,
- - RLRG error system, ∗ ∗ ∗ RLRH error system.

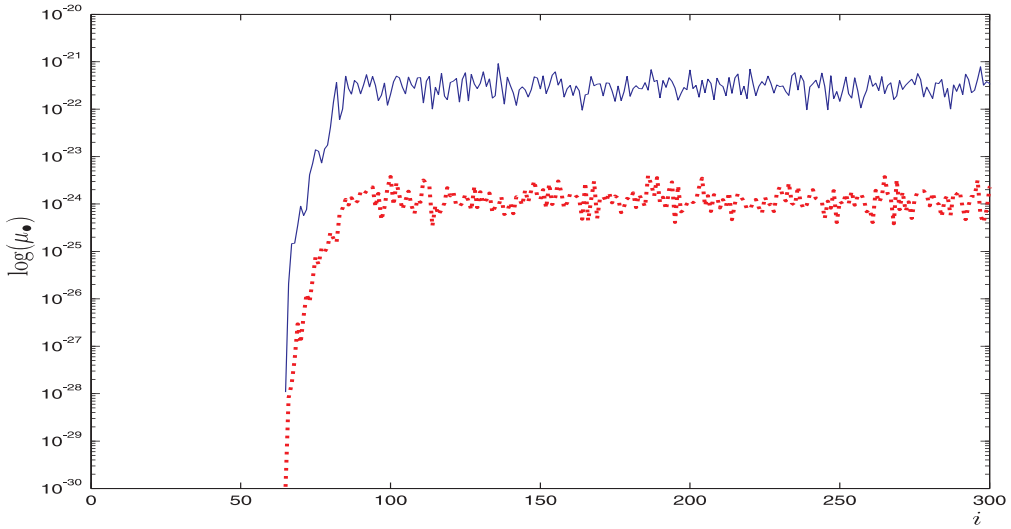


Fig. 6.8. Evolution of the values of $\mu_\bullet$ for ISS 12A model.
— μ_c and ··· μ_o.

6.5 Concluding remarks

In this chapter we propose a new recursive model reduction method based on the Hankel operator. We first study the time-varying case. Subsequently the approach for computing approximate Hankel operators is derived. This approach provides closer results to those obtained by Balanced Truncation with lower computational cost. A bound on the quality of the approximation is given with some numerical examples. This algorithm is the best algorithm for approximating the balanced truncation in terms of accuracy and computational cost. Its cost is $O(N(n+m)(n+p))$, which is only linear in the large dimension N, unlike Balanced Truncation which has a cost which is cubic in the large dimension (i.e., $O(N^3)$). The numerical examples show that this algorithm has very good properties in term of stability, convergence rate and the quality of the approximation.

Despite the obviously desirable features of the Hankel operators approach proposed here, many open questions remain. There are a number of refinements with respect to performance, convergence, and accuracy which require more theoretical and algorithmic analysis. There are particularly two interesting features:

- As for the RLRG algorithm, one needs to have some ideas on the fixed points of the RLRH algorithm. We guess that the fixed point will have the same properties than the fixed point of the RLRG algorithm. Moreover, this fixed point will be close balanced. The fixed point corresponding to the controllability part will be directly related to the fixed point corresponding to the observability part.
- The second feature is about the comparison between the original Hankel map and the Hankel map of the reduced order model. For instance, we just compared the original Hankel map and his dominant block approximation. To compare the two Hankel maps we still need an advanced understanding of the algorithm and its features.

Chapter 7

Second order systems

An important class of physical processes are *structural models*. A structural model is represented in the form of second order differential or difference equations, also called *second order models* [59]. The system dynamics are then typically represented by the degrees of freedom of the system rather than by the system states for a state space description.
A second order system is derived either from physical laws, such as for example Newton's motion laws or Lagrange's equations of motion [89, 8] or from finite-element models [101, 78]. Such systems are well-studied and a significant amount is known about mathematical, physical and geometric structure properties of these systems. These properties play a fundamental role in the understanding of the behaviour of mechanical systems. It also leads to computational methods which take advantage of the structure of the data, for example in ensuring that numerical integration methods conserve energy or momentum.

A second order linear time invariant system is of the form

$$\begin{cases} M\ddot{q}(t) + D\dot{q}(t) + Kq(t) = F^{in}u(t) \\ y(t) = F^{out}q(t) \end{cases}. \tag{7.1}$$

where $u(t) \in \mathbb{R}^m$, $y(t) \in \mathbb{R}^p$, $q(t) \in \mathbb{R}^N$, $F^{in} \in \mathbb{R}^{N\times m}$, $F^{out} \in \mathbb{R}^{p\times N}$, and $M, D, K \in \mathbb{R}^{N\times N}$. Often the physical origin of these systems implies that the matrices M, D and K are symmetric and moreover M is invertible. In most practical cases, m and p are much smaller than N.
For mechanical systems the matrices M, D and K represent, respectively, the *inertia*, *damping* and *stiffness* matrices, $u(t)$ corresponds to the vector of *external* forces, F^{in} is the input distribution matrix, $y(t)$ is the output measurement vector, F^{out} is the output measurement matrix, and $q(t)$ to the vector of *internal generalized coordinates* (see [101] and [78] for more information on such models).

The first equation in 7.1 is a differential equation. One can approximate the first and second differentials in this equation using differences and we obtain an equivalent difference equation representing also a second order system

$$\begin{cases} \hat{M}q_{i+1} + \hat{D}q_i + \hat{K}q_{i-1} = F^{in}u_i \\ y_i = F^{out}q_i \end{cases} \tag{7.2}$$

Table 7.1 shows how to deduce 7.2 from 7.1 using difference schemes, where h is the step size and $q(ih) =: q_i$. Remark that if the matrices M, C and K are symmetric, $\hat{M}$, $\hat{C}$ and $\hat{K}$ will also be symmetric.
The transfer functions associated with the systems (7.1) and (7.2) are[1]

$$T_f(s) \doteq F^{out}P(s)^{-1}F^{in}, \quad \text{and} \quad T_f(z) \doteq F^{out}P(z)^{-1}F^{in} \tag{7.3}$$

where

[1] For the discrete-time case we use the z-transform (Page.13)

$$P(s) \doteq \left(Ms^2 + Ds + K\right), \quad \text{and} \quad P(z) \doteq \left(z\hat{M} + \hat{D} + z^{-1}\hat{K}\right) \tag{7.4}$$

are the *characteristic polynomial matrices*. The zeros of $\det(P(\cdot))$ are also known as the *characteristic frequencies* of the system and play an important role in model reduction. Stability of the system e.g. implies that these zeros must lie in the stability region which is the left half plane for the continuous-time case and the open unit disk of the complex plane (the open disk centered at 0 of radius 1) for the discrete-time case.

$\ddot{q}(t)$	$\dot{q}(t)$	
$\frac{q_{i+1} - 2q_i + q_{i-1}}{h^2}$	$\frac{q_{i+1} - q_i}{h}$	$\hat{M} = \frac{1}{h^2}(M + hD)$ $\hat{D} = \frac{1}{h^2}\left(h^2K - 2M - hD\right)$ $\hat{K} = \frac{1}{h^2}M$
	$\frac{q_i - q_{i-1}}{h}$	$\hat{M} = \frac{1}{h^2}M$ $\hat{D} = \frac{1}{h^2}\left(h^2K - 2M + hD\right)$ $\hat{K} = \frac{1}{h^2}(M - hD)$
	$\frac{q_{i+1} - q_{i-1}}{2h}$	$\hat{M} = \frac{1}{2h^2}(2M + hD)$ $\hat{D} = \frac{1}{h^2}\left(h^2K - 2M\right)$ $\hat{K} = \frac{1}{2h^2}(2M - hD)$

Table 7.1. Relation between differential and difference second order systems.

In civil engineering or aeronautics, the size N of such models is often so high that many analysis and design problems can not be solved anymore within a reasonable computing time. An example is given by the dynamic motion of an aircraft wing in flight, where the typical motions of the wing are large-scale and often relatively simple bending dynamics.
The computational advantages are multiplied when considering, for example, performance evaluation for an aircraft wing under dynamic loading. Here finite element methods are typically used, and performance is checked via Monte-Carlo sampling. Since a large number of repeated simulation trials must be performed, any reduction in the computational costs per simulation can allow a greater exploration of Monte-Carlo space. It is then advisable to construct a reduced order model that nevertheless keeps the "mechanical" structure of the system. Furthermore, the model reduction should preserve the essential features of the systems dynamics (see [82] for some ideas of model reduction techniques which preserve structure of mechanical systems).

We thus need to construct a reduced order model

$$\begin{cases} M_n\ddot{\hat{q}}(t) + D_n\dot{\hat{q}}(t) + K_n\hat{q}(t) = F_n^{in}u(t) \\ \hat{y}(t) = F_n^{out}\hat{q}(t) \end{cases}, \tag{7.5}$$

or equivalently

$$\begin{cases} \hat{M}_n\hat{q}_{i+1} + \hat{D}_n\hat{q}_i + \hat{K}_n\hat{q}_{i-1} = \hat{F}_n^{in}u_i \\ \hat{y}_i = \hat{F}_n^{out}\hat{q}_i \end{cases}, \tag{7.6}$$

where $\hat{y}_i \in \mathbb{R}^p$, $\hat{q}_i \in \mathbb{R}^n$, $\hat{F}_n^{in} \in \mathbb{R}^{n\times m}$, $\hat{F}_n^{out} \in \mathbb{R}^{p\times n}$, and $\hat{M}_n, \hat{D}_n, \hat{K}_n \in \mathbb{R}^{n\times n}$, such that its transfer function is "close" to the original transfer function. We focus on methods based on a projection of dynamics (§.2.1).

Since Equations 7.2 can be considered as a particular case of linear time-invariant systems, we may consider its corresponding (linearized) state-space model (see §.7.1.1) and apply the techniques of model reduction to state-space models. In doing so, the reduced-order system is generally not of the same type

anymore and the symmetry and structure of the data are lost. Since from a physical point of view it makes sense to impose the reduced-order system to be of the same type, one has to add some constraints on the projection matrices to keep the original structure of the data, if we want to use state-space approaches. But we can handle directly the second order system using *modal approximation*. The main idea is based on a selection of some characteristic frequencies λ_i lying for example in a particular frequency range or as close as possible to the unit circle. This selection involves a selection of left and right eigenvectors x_i and y_i of $P(\cdot)$ solution of the generalized eigenvalue problem

$$\begin{bmatrix} -\lambda_i\hat{K} & \hat{K} \\ \hat{K} & \lambda_i\hat{M}+\hat{D} \end{bmatrix} \begin{bmatrix} x_i \\ \lambda_i x_i \end{bmatrix} = 0, \quad \begin{bmatrix} y_i^T & \lambda_i y_i^T \end{bmatrix} \begin{bmatrix} -\lambda_i\hat{K} & \hat{K} \\ \hat{K} & \lambda_i\hat{M}+\hat{D} \end{bmatrix} = 0$$

which are equivalent to $P(\lambda_i)x_i = 0$ and $y_i^T P(\lambda_i) = 0$. A selection of n of these left and right eigenvectors are then put in the $N \times n$ matrices X and Y. Then the matrices of the reduced order model (7.5) are obtained as follows

$$\hat{M}_n = Y^T\hat{M}X,\ \hat{D}_n = Y^T\hat{D}X,\ \hat{K}_n = Y^T\hat{K}X,\ \hat{F}_n^{in} = Y^T\hat{F}^{in}, \text{ and } \hat{F}_n^{out} = \hat{F}^{out}X.$$

A critical drawback of this method is that we choose n frequencies λ_i but we automatically obtain n other frequencies $\bar{\lambda}_i$ (complex conjugate of λ_i) which we do not control. These frequencies can make our reduced model unstable [31, 120].

As we have seen before, model reduction methods are often based on Gramians, and the question now is : can we do the same thing with second order systems? The answer is not easy, as the Gramians correspond to some energy functions for state-space systems, so we have to correctly define Gramians for second order systems. Meyer and Srinivasan have shown in [90] that one can define an equivalent notion of Gramians for second order systems called *second order Gramians*. And they have presented an adaptation of Balanced Truncation for second order linear systems. The main idea is to use second order Gramians and adapt the related optimization problems (see Page.34) to obtain the desired form (§.7.3).
There is also an adaptation of Krylov-subspace techniques for second order systems, which was presented by Su and Craig. Jr [118]. The idea is based on an adaptation of the notion of Krylov-subspace to second order systems (§.7.4).

In the sequel we present a new method which uses a variant of Balanced Truncation applied to second order Gramians. The key idea is to approximate the Gramians related to the corresponding linearized state space model by a block diagonal approximation. It was shown in [31] that this is sufficient to obtain a reduced-order model with the same structure.
We will see also how to adapt the RLRG and RLRH algorithms to this kind of system.

But, let us first see how to linearize a second order system to obtain a state space system.

7.1 Modelling and dynamic projection of second order system

7.1.1 Modelling of second order system

We consider the following second order system of differential equations (7.1)

$$\begin{cases} M\ddot{q}(t) + D\dot{q}(t) + Kq(t) = F^{in}u(t), \\ y(t) = F^{out}q(t). \end{cases} \tag{7.7}$$

This system can be rewritten in state space form using $x(t) = \begin{bmatrix} q(t)^T & \dot{q}(t)^T \end{bmatrix}^T$, and we obtain the generalized state-space system

$$\begin{cases} \begin{bmatrix} I & 0 \\ 0 & M \end{bmatrix} \dot{x}(t) = \begin{bmatrix} 0 & I \\ -K & -D \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ F^{in} \end{bmatrix} u(t), \\ y(t) = \begin{bmatrix} F^{out} & 0 \end{bmatrix} x(t). \end{cases} \tag{7.8}$$

Since M is invertible, we can transform it to standard state-space form

$$\begin{cases} \dot{x}(t) = \begin{bmatrix} 0 & I \\ -M^{-1}K & -M^{-1}D \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ M^{-1}F^{in} \end{bmatrix} u(t), \\ y(t) = \begin{bmatrix} F^{out} & 0 \end{bmatrix} x(t). \end{cases} \tag{7.9}$$

For a discrete-time system (7.2) we can also use $x_i = \begin{bmatrix} q_{i-1}^T & q_i^T \end{bmatrix}^T$, and we obtain

$$\begin{cases} \begin{bmatrix} I & 0 \\ 0 & \hat{M} \end{bmatrix} x_{i+1} = \begin{bmatrix} 0 & I \\ -\hat{K} & -\hat{D} \end{bmatrix} x_i + \begin{bmatrix} 0 \\ \hat{F}^{in} \end{bmatrix} u_i, \\ y_i = \begin{bmatrix} 0 & \hat{F}^{out} \end{bmatrix} x_i. \end{cases} \tag{7.10}$$

If $\hat{M}$ is invertible[2], we can transform it to standard state-space form

$$\begin{cases} x_{i+1} = \begin{bmatrix} 0 & I \\ -\hat{M}^{-1}\hat{K} & -\hat{M}^{-1}\hat{D} \end{bmatrix} x_i + \begin{bmatrix} 0 \\ \hat{M}^{-1}\hat{F}^{in} \end{bmatrix} u_i, \\ y_i = \begin{bmatrix} 0 & \hat{F}^{out} \end{bmatrix} x_i. \end{cases} \tag{7.11}$$

One easily checks that the transfer functions of the systems (7.8) and (7.10) are given respectively by (7.4).

If there is some symmetry in the system and we want to have a symmetric state-space system we can use the following linearization.

$$\begin{cases} E\dot{x}(t) = Ax(t) + Bu(t), \\ y(t) = Cx(t), \end{cases} \tag{7.12}$$

where

$$E = \begin{bmatrix} D & M \\ M & 0 \end{bmatrix}, \; A = \begin{bmatrix} -K & 0 \\ 0 & M \end{bmatrix}, \; B = \begin{bmatrix} F^{in} \\ 0 \end{bmatrix}, \quad \text{and} \quad C = \begin{bmatrix} F^{out} & 0 \end{bmatrix}. \tag{7.13}$$

This is very useful if all matrices M, K and D are symmetric. In this case, we obtain two matrices A and E which are also symmetric.

7.1.2 Dynamic projection of second order system

As we are interested in the projection dynamics based model reduction, we want to construct two projection matrices $X, Y \in \mathbb{R}^{2N \times 2n}$. We consider the following partition, where each block is $N \times n$:

$$X = \begin{bmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{bmatrix}, \quad Y = \begin{bmatrix} Y_{11} & Y_{12} \\ Y_{21} & Y_{22} \end{bmatrix}. \tag{7.14}$$

Starting from any general linearization of a second order system (§.7.1.1), to preserve the special structure of data we have to choose X and Y so that

$$\begin{gathered} Y^T \begin{bmatrix} I & 0 \\ 0 & M \end{bmatrix} X = \begin{bmatrix} T_1 & 0 \\ 0 & M_n \end{bmatrix}, \quad Y^T \begin{bmatrix} 0 & I \\ -K & -D \end{bmatrix} X = \begin{bmatrix} 0 & T_2 \\ -K_n & -D_n \end{bmatrix}, \\ Y^T \begin{bmatrix} 0 \\ F^{in} \end{bmatrix} = \begin{bmatrix} 0 \\ F_n^{in} \end{bmatrix}, \quad \text{and} \quad \begin{bmatrix} F^{out} & 0 \end{bmatrix} X = \begin{bmatrix} F_n^{out} & 0 \end{bmatrix}, \end{gathered} \tag{7.15}$$

[2] We assume M to be invertible so $\hat{M}$ will be invertible if $h|\frac{\lambda_{\max}(D)}{\lambda_{\min}(M)}| < 1$.

where $T_i,\ i = 1, 2$ are invertible matrices. Sufficient conditions to obtain this for all M, D, K, F^{in} and F^{out}, are to choose X and Y block-diagonal, i.e.,

$$X_{12} = 0, \quad X_{21} = 0, \quad Y_{12} = 0, \quad Y_{21} = 0, \tag{7.16}$$

provided $T_1 \doteq Y_{11}^T X_{11}$ and $T_2 \doteq Y_{11}^T X_{22}$ are invertible [31]. In such a case,

$$M_n = Y_{22}^T M X_{22},\ D_n = Y_{22}^T D X_{22},\ K_n = Y_{22}^T K X_{11},$$
$$F_n^{in} = Y_{22}^T F^{in},\ F_n^{out} = F^{out} X_{11}.$$

In order to obtain a reduced order model in standardized form, it suffices to choose $\tilde{X} \doteq X.T^{-1}$ where $T = \mathrm{diag}\{T_1, T_2\}$. The reduced order model equations then have $T_1 = T_2 = I_N$ and

$$M_n = Y_{22}^T M \tilde{X}_{22},\ D_n = Y_{22}^T D \tilde{X}_{22},\ K_n = Y_{22}^T K \tilde{X}_{11},$$
$$F_n^{in} = Y_{22}^T F^{in},\ F_n^{out} = F^{out} \tilde{X}_{11}.$$

In the sequel we show how to compute such projection matrices from Gramians. In the sequel we consider the continuous-time case in order to compare our new method to the literature.

7.2 Second-Order Balanced Truncation

The main idea of Second-Order Balanced Truncation (SOBT) is to dissociate the blocks of Gramians corresponding to the *positions* from those corresponding to the *velocities* as follows. First, we need to define *two* pairs of $N \times N$ second order Gramians that have the same features as traditional Gramians, i.e., they have to change according to similarity transformations, and they must have some energetic interpretation. (Only then a balance and truncate process makes sense). The first pair $(\mathcal{G}_c^{pos}, \mathcal{G}_o^{pos})$ will correspond to an energy optimization problem depending only on the positions $q(t)$ and not on the velocities $\dot{q}(t)$. Reciprocally, the second pair $(\mathcal{G}_c^{vel}, \mathcal{G}_o^{vel})$ will be associated to an optimization problem depending only on the velocities $\dot{q}(t)$ and not on the positions $q(t)$. By analogy to the first order case, the Gramians $\mathcal{G}_o^{pos}$ and $\mathcal{G}_o^{vel}$ will be defined from the adjoint systems[3].
After these definitions we then come to the balancing part of the method. For this we transform to a balanced coordinate system in which the second order Gramians are equal and diagonal :

$$\bar{\mathcal{G}}_c^{\,pos} = \bar{\mathcal{G}}_o^{\,pos} = \Sigma^{pos}, \quad \bar{\mathcal{G}}_c^{\,vel} = \bar{\mathcal{G}}_o^{\,vel} = \Sigma^{vel}.$$

Their diagonal values will enable us to point out what the *important* positions and the *important* velocities are, i.e., those with (hopefully) large effect on the I/O map. Hence to get a reduced second order model we keep only the part of the system that depends on these variables. This is the truncation part of the method.

Let us first define a pair of second order Gramians measuring the contribution of the position coordinates (independently of the velocities) with respect to the I/O map. A natural optimization problem (see [90]) associated with the second order form is the following

$$\min_{\dot{q}_0 \in \mathbb{R}^n} \min_{u(t)} J(u(t), -\infty, 0), \tag{7.17}$$

[3]

- Here we consider the system given by (7.9);
- The adjoint system of $\{A, B, C\}$ is given by [28]

$$\begin{cases} \dot{\tilde{x}}(t) = -A^T \tilde{x}(t) - C^T \tilde{y}(t) \\ \tilde{u}(t) = B^T \tilde{x}(t) \end{cases}.$$

subject to

$$M\ddot{q}(t) + D\dot{q}(t) + Kq(t) = F^{in}u(t), \quad q(0) = q_0,$$

where the objective function J is given by

$$J(v(t), a, b) \doteq \int_a^b v(t)^* v(t) dt.$$

One easily sees (a proof is given in [90]) that the optimum is $q_0^T {\mathcal{G}_{c11}}^{-1} q_0$, where $\mathcal{G}_{c11}$ is the $N \times N$ left upper block of $\mathcal{G}_c$ the solution of

$$A\mathcal{G}_c + \mathcal{G}_c A^* + BB^* = 0. \tag{7.18}$$

The solution of the dual problem will correspond to $q_0^T {\mathcal{G}_{o11}}^{-1} q_0$, where $\mathcal{G}_{o11}$ is the $N \times N$ left upper block of $\mathcal{G}_o$ the solution of

$$A^*\mathcal{G}_o + \mathcal{G}_o A + C^*C = 0. \tag{7.19}$$

Under the change of coordinates $q(t) = \Phi\bar{q}(t)$, it is immediate to verify that this pair of Gramians undergoes a congruence transformation :

$$\left(\bar{\mathcal{G}}_{c11}, \bar{\mathcal{G}}_{o11}\right) = \left(\Phi^{-1}\mathcal{G}_{c11}\Phi^{-T}, \Phi^T\mathcal{G}_{o11}\Phi\right).$$

This implies that there exists a new coordinate system such that both $\mathcal{G}_{c11}$ and $\mathcal{G}_{o11}$ are equal and diagonal. Their energetic interpretation is given by looking at the underlying optimization problem. In (7.17), one looks for the minimal necessary energy to reach the given position q_0 in a such a way that this energy would concentrate on the positions $q(t)$ and not spread in the velocities $\dot{q}(t)$. Hence these Gramians really describe how the I/O energy is distributed among the positions.

Analogously, let us define a pair of second order Gramians that would give the contribution of the velocities with respect to the I/O map. The optimization problem associated is the following

$$\min_{q_0 \in \mathbb{R}^N} \min_{u(t)} \; J(u(t), -\infty, 0) \tag{7.20}$$

subject to

$$M\ddot{q}(t) + D\dot{q}(t) + Kq(t) = F^{in}u(t), \quad \dot{q}(0) = \dot{q}_0.$$

By exactly following the same reasoning as in [90] for the optimization problem (7.17), one can show that the solution of (7.20) is $\dot{q}_0^T {\mathcal{G}_{c22}}^{-1} \dot{q}_0$, where $\mathcal{G}_{c22}$ is the $N \times N$ right lower block of $\mathcal{G}_c$. The solution of the dual problem will correspond to $\dot{q}_0^T {\mathcal{G}_{o22}}^{-1} \dot{q}_0$, where $\mathcal{G}_{o22}$ is the $N \times N$ right lower block of $\mathcal{G}_o$. Again under the change of coordinates $q(t) = \Phi\bar{q}(t)$ one can check that this pair of Gramians undergoes a congruence transformation. Here too the energetic interpretation is given by looking at the underlying optimization problem. In (7.20), one looks for the minimal necessary energy to reach the given velocity $\dot{q}_0$ in a such a way that this energy would concentrate on the velocities and not spread in the positions. Hence these Gramians really describe how the I/O energy is distributed among the velocities.

The conclusion is that these second order Gramians are good candidates for our problem. We make thus the choice :

$$(\mathcal{G}_c^{pos}, \mathcal{G}_o^{pos}) = (\mathcal{G}_{c11}, \mathcal{G}_{o11}) \quad \text{and} \quad \left(\mathcal{G}_c^{vel}, \mathcal{G}_o^{vel}\right) = (\mathcal{G}_{c22}, \mathcal{G}_{o22}). \tag{7.21}$$

In the new model reduction technique that we propose here, we want to be able to balance both pairs of second order Gramians at the same time, and this is not possible with a change of coordinates of the type $q(t) = \Phi\bar{q}(t)$. For these reasons we work in a state-space context, starting with the system (7.9). The method SOBT proceeds then as follows :

1. Gramians computation :
   Compute both pairs of second order Gramians $(\mathcal{G}_c^{pos},\ \mathcal{G}_o^{pos})$ and $(\mathcal{G}_c^{vel},\ \mathcal{G}_o^{vel})$ and put them into block-diagonal matrices :
   $$\mathcal{X} = \begin{bmatrix} \mathcal{G}_c^{pos} & 0 \\ 0 & \mathcal{G}_c^{vel} \end{bmatrix}, \quad \mathcal{Y} = \begin{bmatrix} \mathcal{G}_o^{pos} & 0 \\ 0 & \mathcal{G}_o^{vel} \end{bmatrix}.$$
   (Notice that these are the block diagonal parts of $\mathcal{G}_c$ and $\mathcal{G}_o$, respectively.)
2. Balancing :
   Compute the similarity transformation
   $$S = \begin{bmatrix} \Phi_1 & 0 \\ 0 & \Phi_2 \end{bmatrix}$$
   making $\mathcal{X}$ and $\mathcal{Y}$ equal and diagonal. The transformed system is then
   $$\begin{cases} \dot{\bar{x}}(t) = \begin{bmatrix} \Phi_1^{-1} & 0 \\ 0 & \Phi_2^{-1} \end{bmatrix} \begin{bmatrix} 0 & I \\ -M^{-1}K & -M^{-1}D \end{bmatrix} \begin{bmatrix} \Phi_1 & 0 \\ 0 & \Phi_2 \end{bmatrix} \bar{x}(t) + \begin{bmatrix} 0 \\ \Phi_2^{-1}M^{-1}F^{in} \end{bmatrix} u(t) \\ y(t) = \begin{bmatrix} F^{out}\Phi_1 & 0 \end{bmatrix} \bar{x}(t). \end{cases} . \tag{7.22}$$
3. Truncation :
   Partition $\bar{x}$ as $\begin{bmatrix} \bar{q}_+^T & \bar{q}_-^T & \dot{\bar{q}}_+^T & \dot{\bar{q}}_-^T \end{bmatrix}^T$ where $\bar{q}_+$ (resp. $\dot{\bar{q}}_+$) of dimension n corresponds to the n largest eigenvalues $\mathcal{G}_c^{pos}\mathcal{G}_o^{pos}$ (resp. $\mathcal{G}_c^{vel}\mathcal{G}_o^{vel}$), i.e., to the most controllable *and* observable positions (resp. velocities) with respect to (7.21), and keep the part of the system (7.22) that only depends on the variables $\bar{q}_+(t)$, $\dot{\bar{q}}_+(t)$. This yields the n-dimensional reduced second order system given by the matrices :
   $$\begin{aligned} M_n &= I_n, \\ D_n &= W^{-1}[\Phi_2^{-1}M^{-1}D\Phi_2]_{11}W, \\ K_n &= W^{-1}[\Phi_2^{-1}M^{-1}K\Phi_1]_{11}, \\ F_n^{in} &= W^{-1}[\Phi_2^{-1}M^{-1}F^{in}]_{1:}, \\ F_n^{out} &= [M^{-1}F^{out}\Phi_1]_{:1}\} \end{aligned}$$
   where $W^{-1} = [\Phi_1^{-1}\Phi_2]_{11}$.

Remark 7.1. In practice we do not compute explicitly the state-space realization (7.22). Rather we compute the following dominant left and right invariant subspaces $X_1, X_2, Y_1, Y_2 \in \mathbb{R}^{N\times n}$ such that $Y_1^T X_1 = I_n$, $Y_2^T X_2 = I_n$ and

$$\mathcal{G}_o^{pos}\mathcal{G}_c^{pos}Y_1 = Y_1\Lambda^{pos+}\ , \mathcal{G}_c^{pos}\mathcal{G}_o^{pos}X_1 = X_1\Lambda^{pos+}, \tag{7.23}$$
$$\mathcal{G}_o^{vel}\mathcal{G}_c^{vel}Y_2 = Y_2\Lambda^{vel^+}\ , \mathcal{G}_c^{vel}\mathcal{G}_o^{vel}X_2 = X_2\Lambda^{vel^+}, \tag{7.24}$$

where Λ^{pos+} is a $n \times n$ matrix containing the n largest eigenvalues of $\mathcal{G}_c^{pos}\mathcal{G}_o^{pos}$ and Λ^{vel^+} is a $n \times n$ matrix containing the n largest eigenvalues of $\mathcal{G}_c^{vel}\mathcal{G}_o^{vel}$.
Defining $\tilde{X}_2 \doteq X_2 W$ and $\tilde{Y}_2 \doteq Y_2 W^{-T}$ where $W^{-1} = (Y_1^T X_2)$, the reduced second order model is given by the matrices :

$$\begin{aligned} M_n &= I_n, \\ D_n &= \tilde{Y}_2^T M^{-1} D \tilde{X}_2, \\ K_n &= \tilde{Y}_2^T M^{-1} K X_1, \\ F_n^{in} &= \tilde{Y}_2^T M^{-1} F^{in}, \\ F_n^{out} &= F^{out} X_1. \end{aligned}$$

Remark 7.2. This method can easily be extended to k-th order linear time-invariant systems. Indeed one can define k pairs of $N \times N$ Gramians exactly in the same way we did for 2-nd order systems, i.e., from optimization problems. The i-th pair contains information about the distribution of the I/O energy among the i-th derivative variables $q^{(i)}(t)$.
One sees easily that this pair is given by $(\mathcal{G}_{cii}, \mathcal{G}_{oii})$, where $\mathcal{G}_{cii}$ and $\mathcal{G}_{oii}$ are the i-th $N \times N$ diagonal block of the $kN \times kN$ Gramians $\mathcal{G}_c$ and $\mathcal{G}_o$ ((7.18) and (7.19)).
Then considering a state-space realization of the system, one balances these k pairs simultaneously using a k-blocks diagonal transformation, in order to be able to determine the important part of each "component" $q^{(i-1)}(t)$ of the state $x(t)$. One obtains then a reduced model by keeping the subsystem that only depends on these variables.

Remark 7.3. It is shown in [31] that one can avoid to first transform M to the identity matrix, by working directly on a generalized state-space model for second order systems. All the formulas derived here essentially extend to that case and also allow to exploit the symmetry of the matrices M, D, K, if present. If moreover we have $F^{in} = (F^{out})^T$. One can then obtain reduced second order systems that still preserve that symmetry (we refer to [31] for more details). In this case the transfer function is symmetric, and the second order system is said to be *symmetric*.

7.3 Comparison with the Mayer and Srinivasan approach

In this section, we compare our method (SOBT) with the method presented in [90]. This method produces a second order system and is also inspired from a balanced truncation technique. The main ideas of [90] are the following. First one has to define second order Gramians. To do so the following optimization problems analogous to (7.17) and (7.20) are proposed. The first problem is

$$\min_{\dot{q}_0 \in \mathbb{R}^N} \min_{u(t)} J(u(t), -\infty, 0),$$

subject to

$$M\ddot{q}(t) + D\dot{q}(t) + Kq(t) = F^{in}u(t), \quad q(0) = q_0$$

for which the optimum is proved to be $q_0^T {\mathcal{G}_{c11}}^{-1} q_0$, where $\mathcal{G}_{c11}$ is the $N \times N$ left upper block of $\mathcal{G}_c$.
The second problem is

$$\min_{u(t)} J(u(t), -\infty, 0)$$

subject to

$$M\ddot{q}(t) + D\dot{q}(t) + Kq(t) = F^{in}u(t), \quad q(0) = q_0 \quad \dot{q}(0) = 0,$$

for which the optimum is proved to be $q_0^T (S_n(\mathcal{G}_{c22}))^{-1} q_0$ where $S_n(\mathcal{G}_{c22})$ is the Schur complement of the $N \times N$ right bottom block of $\mathcal{G}_c$. Then based on this, two pairs of second order Gramians are defined. The second order "free velocity" Gramians are

$${\mathcal{G}_c}^{fv} \doteq \mathcal{G}_{c11}, \quad \text{and} \quad {\mathcal{G}_o}^{fv} \doteq \mathcal{G}_{o11},$$

and the second order "zero velocity" Gramians are

$${\mathcal{G}_c}^{zv} \doteq \mathcal{G}_{c22}, \quad \text{and} \quad {\mathcal{G}_o}^{zv} \doteq \mathcal{G}_{o22}.$$

The reduction process is then given below. We give here only the free velocity reduction method since the zero velocity version follows by analogy.

1. By a change of coordinates (preserving the symmetry of the data if any), put the matrix M equal to the identity. The second order model is then $\{I, M^{-1}D, M^{-1}K, M^{-1}F_M^{in}, F^{out}\}$.
2. Compute the similarity transformation $q(t) = \Phi\bar{q}(t)$ such that

$$\Phi^{-1}{\mathcal{G}_c}^{fv}\Phi^{-T} = \Sigma^{fv} = \Phi^T{\mathcal{G}_o}^{fv}\Phi,$$

   where Σ^{fv} is a positive diagonal matrix with diagonal values sorted in decreasing order. Define $V \in \mathbb{R}^{N\times n}$ to be the first n columns of Φ.
3. The reduced system is then given by

$$M_n = V^TV, D_n = V^TM^{-1}DV, K_n = V^TM^{-1}KV,$$
$$F_n^{in} = V^TM^{-1}F^{in}, F_n^{out} = F^{out}V.$$

We point out some drawbacks of this method.
From $\{I, M^{-1}D, M^{-1}K, M^{-1}F^{in}, F^{out}\}$ a balanced realization

$$\{\Phi, M^{-1}D\Phi, M^{-1}K\Phi, M^{-1}F^{in}, F^{out}\Phi\}$$

is first computed with respect to free velocity Gramians. Truncation (i.e., selection of $n \ll N$ coordinates) is performed on the system matrices after multiplication by Φ^T ! So the reduced model is given by

$$\{[\Phi^T\Phi]_{11}, [\Phi^TM^{-1}D\Phi]_{11}, [\Phi^TM^{-1}K\Phi]_{11}, [\Phi^TM^{-1}F^{in}]_{1:}, [F^{out}\Phi]_{:1}\}. \tag{7.25}$$

It is not clear that the above truncation indeed selects the dominant state vectors transferring input energy to output energy. One would rather have expected a left multiplication by Φ^{-1} to normalize the mass matrix to the identity, followed by a truncation :

$$\{I, [\Phi^{-1}M^{-1}D\Phi]_{11}, [\Phi^{-1}M^{-1}K\Phi]_{11}, [\Phi^{-1}M^{-1}F^{in}]_{1:}, [M^{-1}F^{out}\Phi]_{:1}\}. \tag{7.26}$$

This modified method would be equivalent to our method when imposing $\Phi_2 = \Phi_1$ (and hence "freezes" one degree of freedom). Obviously the motivation for multiplying the equations by Φ^T before truncating is to obtain a reduced model that preserves the symmetry properties of the original system. But a clear motivation from the point of view of the projection error is lacking.
A second drawback is that, with the method proposed in [90], one implicitly assumes that influential velocities are also influential positions, and again there is no reason why this should give good results since one does not use all degrees of freedom at hand. The last drawback lies in the definition of the "zero velocity" Gramians $\mathcal{G}_c^{zv}, \mathcal{G}_o^{zv}$. Contrarily to the "free velocity" Gramians, the definition of $\mathcal{G}_c^{zv}, \mathcal{G}_o^{zv}$ has not been well justified from an energetic point of view. As illustrated in the test examples of the next section, we believe that these drawbacks have a negative effect on the approximation error of that approach.

7.4 Krylov-subspace technique for second order systems

In this section, we discuss a Krylov-subspace technique that produces a reduced-order model of second order form [118, 117, 119]. The key idea is the following. In view of (7.13), the desired Krylov subspace (see §.1.1.3 and §.2.3) for the construction of the projection matrices is

$$span\{\tilde{B}, (A^{-1}E)\tilde{B}, (A^{-1}E)^2\tilde{B}, \ldots, (A^{-1}E)^{n-1}\tilde{B}\},$$

where $\tilde{B} \doteq -A^{-1}\left[\begin{array}{cc} B & C^T \end{array}\right]$ (of course we have assumed that A is nonsingular). Here, the linearization considered is (7.12).

Let us set

$$R_j = \begin{bmatrix} R_j^d \\ R_j^v \end{bmatrix} \doteq (-A^{-1}E)^j \tilde{B},$$

where R_j^d is the vector of length N corresponding to the displacement portion of the vector R_j, and R_j^v is the vector of length N corresponding to the velocity portion of the vector R_j (see [118]). Then, in view of the structure of the matrices A and E, we have

$$\begin{bmatrix} R_j^d \\ R_j^v \end{bmatrix} = (-A^{-1}E)^j \begin{bmatrix} R_{j-1}^d \\ R_{j-1}^v \end{bmatrix} = \begin{bmatrix} K^{-1}DR_{j-1}^d + K^{-1}MR_{j-1}^v \\ -R_{j-1}^d \end{bmatrix}.$$

Note that the j^{th} velocity-portion vector R_j^v is the same (up to its sign) as the $(j-1)^{st}$ displacement-portion vector R_{j-1}^d. In other words, the second portion R_j^v of R_j is the "one-step" delay of the first portion R_{j-1}^d of R_j. This suggest that one may simply choose

$$span\{R_0^d, R_1^d, R_2^d, \ldots, R_{n-1}^d\} \tag{7.27}$$

as the projection subspace used to construct the projection matrices.

The resulting procedure can be summarized as follows[4].

Algorithm 8 The Su and Craig algorithm.

• Initialization
Set $R_0^d = K^{-1}\left[\, F^{in} \;\; (F^{out})^T \,\right]$, $R_0^v = 0$,
Compute the SSVD $U_0\Sigma_0 V_0^T = (R_0^d)^T K R_0^d$,
$Q_0^d = R_0^d U_0 \Sigma_0^{-1/2}$, and $Q_0^v = 0$.
• Arnoldi loop
for $j = 1 : n-1$ **do**
  $R_j^d = K^{-1}(DQ_{j-1}^d + MQ_{j-1}^v)$,
  $R_j^v = -Q_{j-1}^d$,
  • Orthogonalization
  for $i = 1 : j$ **do**
    $T_i = (Q_i^d)^T K R_j^d$,
    $R_j^d = R_j^d - Q_i^d T_i$,
    $R_j^v = R_j^v - Q_i^v T_i$,
  end for
  • Normalization
  Compute the SVD $U_0\Sigma_0 V_0^T = (R_0^d)^T K R_0^d$,
  $Q_{j+1}^d = R_j^d U_0 \Sigma_0^{-1/2}$, and
  $Q_{j+1}^v = R_j^v U_0 \Sigma_0^{-1/2}$.
end for

This algorithm produces a projection matrix Q_n which is an orthonormal basis of (7.27). The reduced order model is given by

$$M_n = Q_n^T M Q_n, D_n = Q_n^T D Q_n, K_n = Q_n^T K Q_n,$$
$$F_n^{in} = Q_n^T F^{in}, F_n^{out} = F^{out} Q_n.$$

In [118, 117, 119], a number of advantages of this approach are described, see also [12] for a further discussion of this algorithm and other techniques.

[4] Here, for numerical stability, one may choose to use the Arnoldi process to generate the basis of (7.27), but a Lanczos algorithm can also be employed here (see [118, 117, 119] for more details).

7.5 RLRG and RLRH for second order systems

The RLRG and RLRH algorithms presented in Chapter 5 and in Chapter 6 can be adapted easily for second order systems (at least for discretized systems). The idea is to use the special structure of the matrices A, B and C of these systems to obtain a recurrence involving directly the matrices of the second order system M, D, K, F^{in}, and F^{out}. In this section we show how to do it. Since we have a second order system, one may expect that the procedure will involve a recurrence on two time steps on a matrix representing the dominant subspace. From this matrix, we will construct the projection matrices directly for the second order system.

It is pointed out that the updated version of the dominant subspace for both algorithms (RLRG and RLRH) is obtained via the recursion :

$$S_n(i+1) = \underbrace{\left[\, AS_n(i) \middle| B \,\right]}_{M_1} \begin{bmatrix} V_{c_1} \\ \hline V_{c_2} \end{bmatrix}, \tag{7.28}$$

and

$$R_n(i+1) = \underbrace{\left[\, A^T R_n(i) \middle| C^T \,\right]}_{M_2} \begin{bmatrix} V_{o_1} \\ \hline V_{o_2} \end{bmatrix}, \tag{7.29}$$

where $V_{c_1}, V_{o_1} \in \mathbb{R}^{n\times n}$ come from the SVD of M_1 and M_2, respectively for the RLRG algorithm or from the SVD of $M_2^T M_1$ for the RLRH algorithm (see Algorithms 6 and 7).

For a second order system, one may consider that $S_\bullet(\cdot)$ corresponds in fact to a collection of two consecutive versions of the dominant subspace of the second order system, i.e., for example for $S_n(\cdot)$

$$S_n(i+1) = \begin{bmatrix} S_n^s(i) \\ \hline S_n^s(i+1) \end{bmatrix},$$

(subscript s referring to "second order system"). It follows from this and from (7.28) that we have

$$\begin{bmatrix} S_n^s(i) \\ \hline S_n^s(i+1) \end{bmatrix} = \left[\begin{bmatrix} 0 & I \\ -\hat{M}^{-1}\hat{K} & -\hat{M}^{-1}\hat{D} \end{bmatrix} \begin{bmatrix} S_n^s(i-1) \\ \hline S_n^s(i) \end{bmatrix} \middle| \begin{bmatrix} 0 \\ \hline M^{-1}\hat{F}^{in} \end{bmatrix} \right] \begin{bmatrix} V_1 \\ \hline V_2 \end{bmatrix}. \tag{7.30}$$

Note first that $S_n^s(i)$ on the left hand side is not the same as on the right hand side, but this is an intermediary updating version which may allow us to update the whole subspace for state-space description. If we denote it $S_{c_+}^s(i)$, (7.30) can be rewritten as

$$\begin{cases} S_{c_+}^s(i) &= S_n^s(i)V_1, \\ S_n^s(i+1) &= -\hat{M}^{-1}\hat{K}S_n^s(i-1)V_1 - \hat{M}^{-1}\hat{D}S_n^s(i)V_1 + \hat{M}^{-1}\hat{F}^{in}V_2. \end{cases} \tag{7.31}$$

Here, the first equation is redundant and one may consider just the second one. But, actually this is not correct as it is an update of $S_n(i)$ which will affect not $S_n(i+1)$ but $S_n(i+2)$ in the next step.

For $R_n(\cdot)$, the adaptation is more complicated. We have

$$\begin{bmatrix} R_n^s(i) \\ \hline R_n^s(i+1) \end{bmatrix} = \left[\begin{bmatrix} 0 & -\hat{K}^T\hat{M}^{-T} \\ I & -\hat{D}^T\hat{M}^{-T} \end{bmatrix} \begin{bmatrix} R_n^s(i-1) \\ \hline R_n^s(i) \end{bmatrix} \middle| \begin{bmatrix} 0 \\ \hline (\hat{F}^{out})^T \end{bmatrix} \right] \begin{bmatrix} U_1 \\ \hline U_2 \end{bmatrix}. \tag{7.32}$$

which yields

$$\begin{cases} S_{o_+}^s(i) &= -\hat{K}^T\hat{M}^{-T}R_n^s(i)U_1, \\ R_n^s(i+1) &= R_n^s(i-1)U_1 - \hat{D}^T\hat{M}^{-T}R_n^s(i)U_1 + (\hat{F}^{out})^T U_2. \end{cases} \tag{7.33}$$

Here, the first equation is not redundant and we have to take it into consideration. One can remark also that the first equation will affect the dominant subspace only after two steps.

Using Recurrences 7.31 and 7.33, we obtain after a certain number of iterations, two $N \times n$ matrices R_n and S_n. These matrices are used to construct the projection matrices X and Y (§.7.1.2) as in the approximated Balanced Truncation.

In the sequel we present the two algorithms adapted for second order systems, namely *SRLRG* for *Second-order Recursive Low-Rank Gramian* and *SRLRH* for *Second-order Recursive Low-Rank Hankel*.
To simplify the algorithms let us define the following matrices :

$$M_1(i) = \left[\begin{bmatrix} 0 & I \\ -\hat{M}^{-1}\hat{K} & -\hat{M}^{-1}\hat{D} \end{bmatrix} \begin{bmatrix} S_n(i-1) \\ S_n(i) \end{bmatrix} \middle| \begin{bmatrix} 0 \\ M^{-1}\hat{F}^{in} \end{bmatrix} \right],$$

$$M_2(i) = \left[\begin{bmatrix} 0 & -\hat{K}^T\hat{M}^{-T} \\ I & -\hat{D}^T\hat{M}^{-T} \end{bmatrix} \begin{bmatrix} R_n(i-1) \\ R_n(i) \end{bmatrix} \middle| \begin{bmatrix} 0 \\ (\hat{F}^{out})^T \end{bmatrix} \right].$$

Algorithm 9 The Second order Recursive Low-Rank Gramians algorithm (SRLRG).

- We initialize the $N \times n$ matrices

$$S_n(0) = 0, \quad S_n(1) = 0, \quad R_n(0) = 0, \quad \text{and} \quad R_n(1) = 0,$$

for $i = 1 : \tau$ **do**

- We compute the SSVDs

$$M_1(i) = U_c \Sigma_c V_c^T, \quad \text{and} \quad M_2(i) = U_o \Sigma_o V_o^T.$$

- We use the following decomposition, where $V_{c_1}, V_{o_1} \in \mathbb{R}^{n \times n}$:

$$V_c(:, 1:n) = \begin{bmatrix} V_{c_1} \\ V_{c_2} \end{bmatrix}, \quad \text{and} \quad V_o(:, 1:n) = \begin{bmatrix} V_{o_1} \\ V_{o_2} \end{bmatrix}.$$

- Then we compute the new $S_n(\cdot)$ and $R_n(\cdot)$ using :

$$\begin{aligned} S_{c_+}(i) &= S_{c_+}(i) V_{c_1}, \\ S_n(i+1) &= -\hat{M}^{-1}\hat{K} S_n(i-1) V_{c_1} - \hat{M}^{-1}\hat{D} S_n(i) V_{c_1} + \hat{M}^{-1}\hat{F}^{in} V_{c_2} \end{aligned}$$

and

$$\begin{aligned} S_{o_+}(i) &= -\hat{K}^T\hat{M}^{-T} R_n(i) V_{o_1} \\ R_n(i+1) &= R_n(i-1) V_{o_1} - \hat{D}^T\hat{M}^{-T} R_n(i) V_{o_1} + (\hat{F}^{out})^T V_{o_2}. \end{aligned}$$

end for

- To construct the reduced model, we compute the SSVD

$$S_n(\tau)^T R_n(\tau) = U \Sigma V^T,$$

- And we obtain the projection matrices

$$X = S_n(\tau) U \Sigma^{-1/2}, \quad \text{and} \quad Y = R_n(\tau) V \Sigma^{-1/2}.$$

- The order n approximated second order system is given by :

$$M_n = Y^T M X, \; D_n = Y^T D X, \; K_n = Y^T K X, \; F_n^{in} = Y^T F^{in}, \; F_n^{out} = F^{out} X.$$

Algorithm 10 The Second order Recursive Low-Rank Hankel algorithm (SRLRH).

• We initialize the $N \times n$ matrices

$$S_n(0) = 0, \quad S_n(1) = 0, \quad R_n(0) = 0, \quad \text{and} \quad R_n(1) = 0,$$

for $i = 1 : \tau$ **do**

• We compute the SSVD

$$M_2^T M_1 = U \Sigma V^T.$$

• We use the following decomposition, where $V_1, U_1 \in \mathbb{R}^{n \times n}$:

$$V(:, 1:n) = \begin{bmatrix} V_1 \\ V_2 \end{bmatrix}, \quad \text{and} \quad U(:, 1:n) = \begin{bmatrix} U_1 \\ U_2 \end{bmatrix}.$$

• Then we compute the new $S_n(\cdot)$ and $R_n(\cdot)$ using :

$$\begin{aligned} S_{c_+}(i) &= S_{c_+}(i) V_1, \\ S_n(i+1) &= -\hat{M}^{-1} \hat{K} S_n(i-1) V_1 - \hat{M}^{-1} \hat{D} S_n(i) V_1 + \hat{M}^{-1} \hat{F}^{in} V_2 \end{aligned}$$

and

$$\begin{aligned} S_{o_+}(i) &= -\hat{K}^T \hat{M}^{-T} R_n(i) U_1 \\ R_n(i+1) &= R_n(i-1) U_1 - \hat{D}^T \hat{M}^{-T} R_n(i) U_1 + (\hat{F}^{out})^T U_2 \end{aligned}$$

end for

• To construct the reduced model, we compute the SSVD

$$S_n(\tau)^T R_n(\tau) = \hat{U} \hat{\Sigma} \hat{V}^T.$$

• And we obtain the projection matrices

$$X = S_n(\tau) \hat{U} \hat{\Sigma}^{-1/2}, \quad \text{and} \quad Y = R_n(\tau) \hat{V} \hat{\Sigma}^{-1/2}.$$

• The order n approximated second order system is given by :

$$M_n = Y^T M X, \; D_n = Y^T D X, \; K_n = Y^T K X, \; F_n^{in} = Y^T F^{in}, \; F_n^{out} = F^{out} X.$$

7.6 Numerical results

This section makes a numerical comparison of several procedures presented in this chapter for second order systems. These algorithms are applied to our benchmark models.

In Table 7.2, we compare eight model reduction methods : the classic Balanced Truncation ('BT'), the methods of Mayer (Free Velocity ('FV'), Zero Velocity ('ZV'), and 'M' for the case where we applied free and zero velocity in the same time), and our methods : the Second-Order Balanced Truncation ('SOBT'), the Second-order Recursive Low Rank Gramian ('SRLRG'), and the Second-order Recursive Low Rank Hankel ('SRLRH'). The comparison is made on the basis of the relative reduction error measured according to the $\mathcal{H}_\infty$ norm, i.e., the ratio of the $\mathcal{H}_\infty$ norms of the "error" system $\mathcal{S} - \mathcal{S}_n$ and the full system $\mathcal{S}$, i.e.,

$$\frac{\|\mathcal{S} - \mathcal{S}_n\|_\infty}{\|\mathcal{S}\|_\infty}.$$

In Figures 7.1, 7.2, 7.3, and 7.4 we show the $\sigma_{\max}$-plot of the frequencies of the full system and the error system for each benchmark model.

The method of Su and Craig produces for the CDplayer model an unstable reduced-order model, but in general we have remarked that even if the original system is strictly stable, the reduced-order model obtained via this method moves to a region close to instability. The methods of Meyer seems to work quite well, but in general the method 'SOBT' behaves better. The adaptation of RLRG and RLRH, namely SRLRG and SRLRH yields the best results. Note that we still use a stopping criterion based on the tolerance values given in (§ 4.8.2).

	Building model	CDplayer model	ISS 1R model	ISS 12A model
$\|\mathcal{S}\|_\infty$	0.0053	$2.3198.10^{6}$	0.1159	0.0107
$\frac{\|\mathcal{S}-\mathcal{S}_n^{BT}\|_\infty}{\|\mathcal{S}\|_\infty}$	0.1143	$8.0704.10^{-8}$	0.0013	0.0071
$\frac{\|\mathcal{S}-\mathcal{S}_n^{M}\|_\infty}{\|\mathcal{S}\|_\infty}$	1.0181	1.0000	1.6324	1.0028
$\frac{\|\mathcal{S}-\mathcal{S}_n^{S}\|_\infty}{\|\mathcal{S}\|_\infty}$	1.0182	unstable	1.6324	1.0063
$\frac{\|\mathcal{S}-\mathcal{S}_n^{SOBT}\|_\infty}{\|\mathcal{S}\|_\infty}$	0.0424	1.0000	0.1054	0.9966
$\frac{\|\mathcal{S}-\mathcal{S}_n^{FV}\|_\infty}{\|\mathcal{S}\|_\infty}$	0.4509	1.0000	0.0548	0.5659
$\frac{\|\mathcal{S}-\mathcal{S}_n^{ZV}\|_\infty}{\|\mathcal{S}\|_\infty}$	0.0379	1.0000	0.0513	0.3537
$\frac{\|\mathcal{S}-\mathcal{S}_n^{RLRG}\|_\infty}{\|\mathcal{S}\|_\infty}$	0.4301	$6.8931.10^{-6}$	0.1023	0.9697
$\frac{\|\mathcal{S}-\mathcal{S}_n^{RLRH}\|_\infty}{\|\mathcal{S}\|_\infty}$	0.4320	$1.7.10^{-6}$	0.0979	0.9390

Table 7.2. $\mathcal{H}_\infty$ norm of benchmark models, and the error systems.

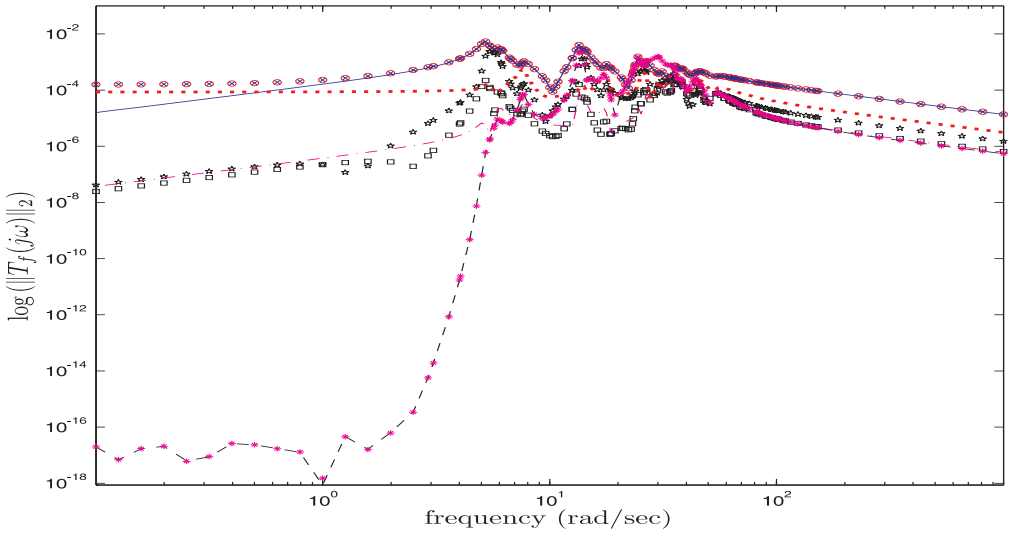


Fig. 7.1. $\sigma_{\max}$-plot of the frequency responses for Building model.

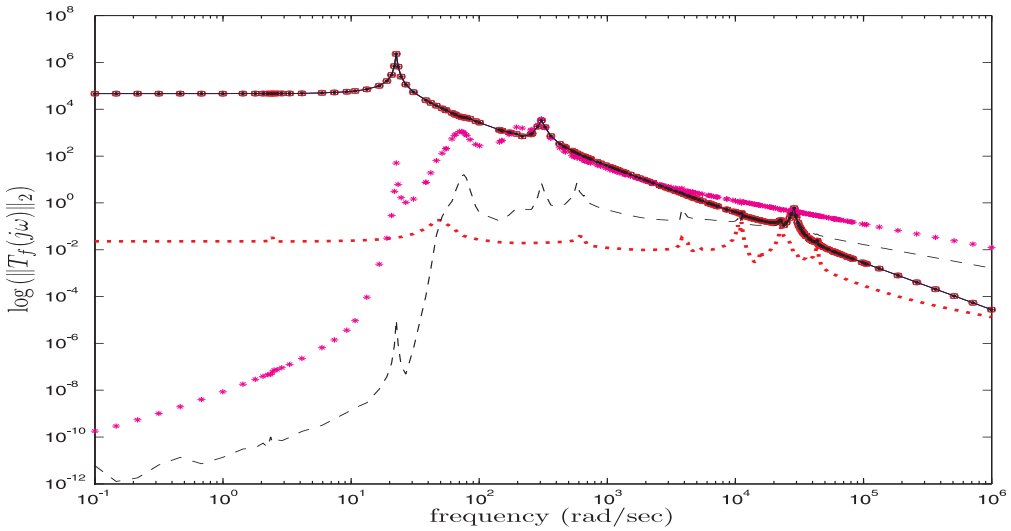


Fig. 7.2. $\sigma_{\max}$-plot of the frequency responses for CDplayer model.

LEGEND : — full model, ··· BT error system, □ SOBT error system, o Mayer error system, × Su error system, ⋆ FV error system, – ZV error system, - - SRLRG error system, ∗ ∗ ∗ SRLRH error system.

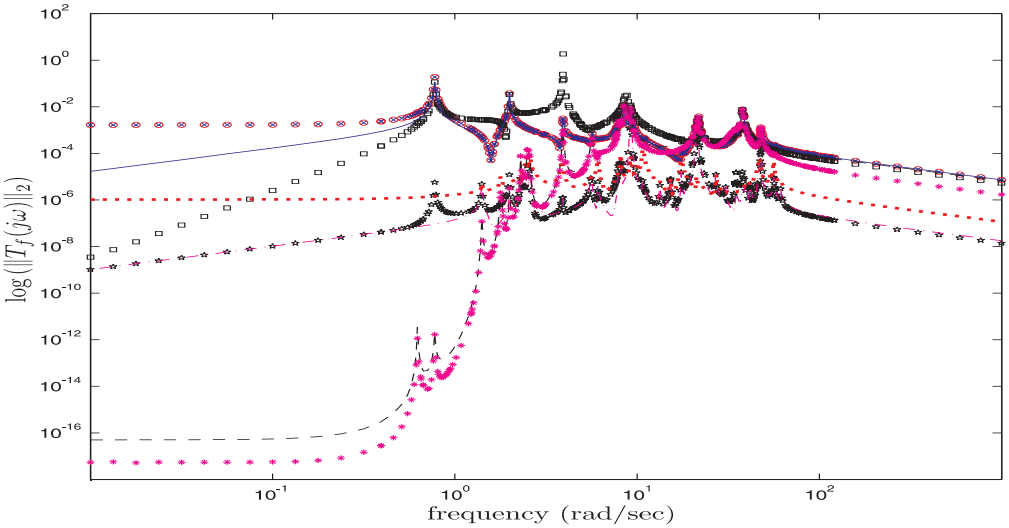


Fig. 7.3. $\sigma_{\max}$-plot of the frequency responses for ISS 1R model.

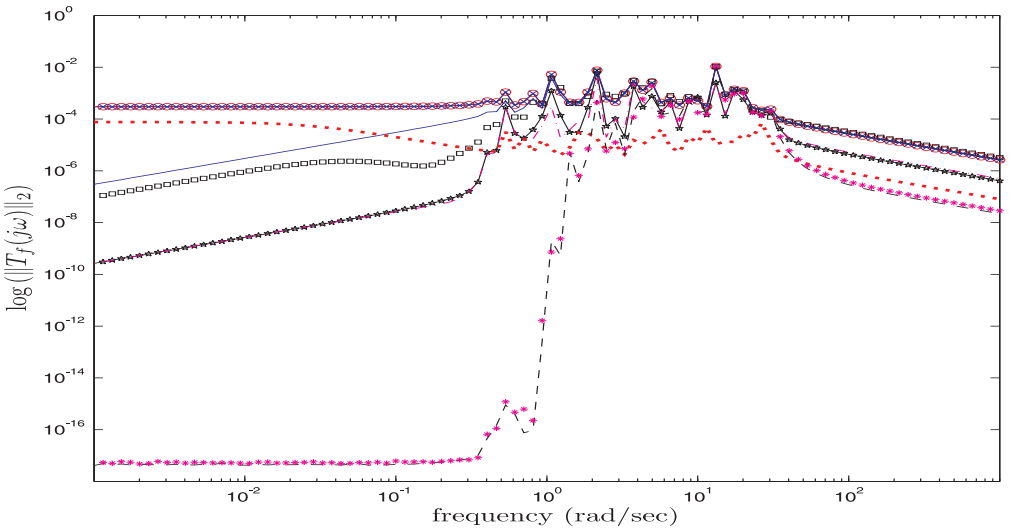


Fig. 7.4. $\sigma_{\max}$-plot of the frequency responses for ISS 12A model.

LEGEND : — full model, ··· BT error system, □ SOBT error system,
o Mayer error system, × Su error system, ⋆ FV error system, – ZV error system,
- - SRLRG error system, ∗ ∗ ∗ SRLRH error system.

7.7 Concluding remarks

In this chapter we have introduced a structure preserving model reduction method for second order systems. It is a balance and truncate method based on the definition of two pairs of $N \times N$ Gramians $(\mathcal{G}_c^{pos}, \mathcal{G}_o^{pos})$, and $(\mathcal{G}_c^{vel}, \mathcal{G}_o^{vel})$. Following the idea of [90], these Gramians are derived from optimization problems, and are shown to contain information about the capacity of positions q_0 and velocities $\dot{q}_0$ to transferring energy between inputs and outputs. Working in a state-space model enables us to balance both pairs of Gramians simultaneously, which allows to determine the n most controllable and observable position components and the n most controllable and observable velocity components, where $n \ll N$. The reduced model is then

obtained by keeping that part of the balanced system that only depends on these variables, which then automatically gives a reduced model of second order type.

We have shown that the method in [90] is not a real balance and truncate method, and that it has a few drawbacks. Numerically speaking our method yields results close to those obtained by the method of Mayer [90, 32], but our method is a veritable balance and truncate method. Finally, the SRLRG and SRLRH seem to be again the best methods.

From a theoretical point of view, there are many open questions. For instance, does SOBT, SRLRG and SRLRH techniques preserve stability?

For the SOBT method, if this turns out to be true, then does there exist a global error bound between the original and the reduced order model depending on the eigenvalues of the product of the Gramians that we neglect, as in the case of the standard balanced truncation ? If this turns out to be false, does there exist other pairs of Gramians that provide a global error bound?

Chapter 8

Conclusions and perspectives

Model reduction of dynamical systems has its roots in many different fields of applied mathematics, even though the methods forming the foundation for model reduction are relatively old. The history of the Singular Value Decomposition, for example, spans more than one century. The earlier algorithms for Krylov-based methods, are fifty years old. It is obvious from the many recent references of this thesis, that the understanding and application of low-rank approximation methods in model reduction is certainly not a closed topic.
In this thesis we tried to propose new ideas of low-rank approximation and we established connections with projection techniques used in model reduction. These ideas were mainly aimed at a time-varying case. The algorithms proposed are based on low-rank approximation of Gramians and Hankel operators. The Hankel operator is defined as the input-output mapping. It is well known that Gramians reflect the energy of input-state and state-output maps, and hence play an important role in the approximation of the Hankel map.

The key idea was the combination of two basic projection based model reduction methods, namely Balanced Truncation and Krylov subspace ideas, to obtain a new method called Approximated Balanced Truncation.
This method has very nice properties. It benefits of the iterative computations inherited from Krylov subspace ideas in the computation of the approximated Gramians. This iterative computations reduce significatively the cost and make benefice of any sparsity in the data. The use of the Balanced Truncation procedure to provide the reduced order model yields bounds on the quality of the approximations.
Actually, the quality of the approximations is depending on how one computes the low-rank approximations of the Gramians. We have presented three algorithms to do this job.

The first algorithm presented, namely the Modified Low-Rank Smith (MLRS) algorithm, was studied in detail. The idea is to exploit the rapid decay of the Gramian eigenvalues to approximate well the square root of the Gramian by a low-rank approximation. This square root can be constructed recursively, and so we have proposed a recursive procedure to compute its low rank approximation. This is done using a window which collects the dominant low-rank part using a local singular value decomposition. This method has a drawback since it depends crucially on the gap between what we keep and what we neglect. To obtain the best results with this approach one has to mix two stopping criteria, namely a tolerance value criterion and a gap criterion. Unfortunately for our benchmark models there is no clear gap, which explains the bad quality of the approximations.

The two other algorithms were mainly meant for time-varying case. The key idea is to compute only a finite window of the Gramians of the provided time-varying system. Then, one can easily come up with a square root version (or Cholesky factor) of the Gramians. The existence of these Cholesky factors is guaranteed by the fact that the Gramians are symmetric and at least positive semi-definite. Rather than computing the exact factors, one keeps a low-rank approximation of the Gramians. These low-rank approximations are computed recursively and updated in such a way that at each step only the n "dominant" vectors are kept. The basic idea for this is to keep at each step the leading n column vectors of the Singular Value Decomposition (SVD). Two ways to do this were described in the presentation of the two Recursive Low-Rank algorithms. The last algorithm namely *Recursive Low-Rank Hankel* (RLRH) approximation method is

the one we recommend. It provides results closer to those of Balanced Truncation at low cost, and it yields directly a balanced approximation of the full Gramians. And so it is a powerful method which will suffer less from a bad balancing of the original system. The *Recursive Low-Rank Gramian* (RLRG) algorithm is also a very interesting method. It also yields results close to those of the Balanced Truncation and of the RLRH method. But RLRG is more expensive than RLRH as we have to "balance" the projector to obtain good results.

The final contributions of this thesis is the work on second order systems. These systems have a special structure which has a physical meaning. In order to keep this structure in the reduced-order model, we presented some new ideas which we compared with previous work in this area. We also showed how to adapt our best algorithms RLRG and RLRH to this kind of systems.

There are still many open questions that one can consider for future research. Despite the obviously desirable features of the Gramians and Hankel approaches proposed here, there are a number of refinements with respect to performance, convergence, and accuracy which can be studied in the future. For instance there is no result on the stability of the reduced-order model obtained via low-rank approximation. And unlike Balanced Truncation there is no global error on the difference between the two transfer functions.

We believe that both time-varying algorithms (RLRG an RLRH) can be generalized to linear infinite dimensional systems. For these systems one will have to iterate on operators $\mathcal{A}$ and $\mathcal{B}$ instead of matrices A and B. An interesting class of linear infinite dimensional systems is the class of *Riesz systems*. This class has the property that the operator $\mathcal{A}$ has a spectral decomposition. This decomposition can be truncated at a certain order to obtain a low-rank approximation of the operator. Of course, we still need to approximate the effect of the operator on an infinite dimensional vector, but we nevertheless think this may improve the quality of the approximation. This raises a very interesting question which is still open as far as we know : "*is it better to reduce and then discretize, or to discretize and then reduce?*".

Another perspective is the generalization of our algorithms to Riccati equations. It is well known that Lyapunov equations are special cases of Riccati equations. The question here is how to use our ideas to find efficiently a low-rank solution of the Riccati equation. These solutions can also be used in a Balanced Truncation procedure.

Recently, many authors try to define Gramians for nonlinear dynamical systems. One can guess that we can establish connections between different ideas presented in this thesis and some popular ideas for nonlinear dynamical systems. Some relevant ideas have been suggested in a survey paper of Van Dooren [128].

Appendix

A- Construction of the Givens rotations G_u and G_v

We show in this appendix the explicit construction of the Givens transformations G_u and G_v (Page.53). Here G_u and G_v are a product of Givens transformations :

$$G_u = G_{u_1} G_{u_2} \dots G_{u_k}, \quad \text{where} \quad G_{u_j} = G_{u_{j,1}} \dots G_{u_{j,n+j-2}} G_{u_{j,n+j-1}}, j = 1:k,$$

$$G_v = G_{v_1} G_{v_2} \dots G_{v_m}, \quad \text{where} \quad G_{v_j} = G_{v_{j,1}} \dots G_{v_{j,n+j-2}} G_{v_{j,n+j-1}}, j = 1:k.$$

Each $G^*_{u_{j,i}}$ is a Givens rotation operating on rows i and $i+1$ only of the column $\hat{U}_2(:,j)$ and each $G_{v_{j,i}}$ is a Givens rotation operating on columns i and $i+1$ only. The rotations $G^*_{u_{j,i}}$ annihilate the consecutive elements of $\hat{U}_2(:,j)$, $j = 1, \dots, m$. We begin by the last column $\hat{U}_2(:,m)$:

$$G^*_{u_{m,1}} \hat{U}_2(:,m) = \begin{bmatrix} \mathbf{0} \\ \times \\ \vdots \\ \times \end{bmatrix}, \quad \dots \quad , \quad \underbrace{G^*_{u_{m,n+m-1}} \dots G^*_{u_{m,2}} G^*_{u_{m,1}}}_{G^*_{u_m}} \hat{U}_2(:,m) = \begin{bmatrix} 0 \\ \vdots \\ \mathbf{0} \\ \times \end{bmatrix},$$

because $\hat{U}_2$ is orthogonal, when we apply G_{u_1} to the whole matrix $\hat{U}_2$ we obtain automatically that the last row is e^*_m, i.e.,

$$G^*_{u_1} \hat{U}_2 = \begin{bmatrix} \times & \dots & \times & 0 \\ \vdots & \ddots & \vdots & \vdots \\ \times & \dots & \times & 0 \\ 0 & \dots & 0 & 1 \end{bmatrix}. \tag{A-1}$$

These rotations are then also applied to the columns of $\hat{R}$ which will destroy its upper triangular form. Therefore the rotations $G_{v_{j,i}}$ are chosen to annulate again the elements $(i, i+1)$ introduced by $G^*_{u_i}$. i.e., in the first step we have :

$$G^*_{u_{m,1}} \hat{R} = \begin{bmatrix} \times & & \dots & \times \\ \times & \times & & \\ 0 & 0 & \ddots & \vdots \\ \vdots & & \ddots & \\ 0 & & \dots & 0 \ \times \end{bmatrix}, \quad G^*_{u_{m,1}} \hat{R} G_{v_{m,1}} = \begin{bmatrix} \times & \dots & \times & 0 \\ \mathbf{0} & \ddots & & \times \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & \times \end{bmatrix}$$

The same holds (recursively) for the successive pairs :

$$G^*_{u_{m,i+1}} G^*_{u_{m,(i)}} \hat{R} G_{v_{m,(i)}} = \begin{bmatrix} \times & \dots & \times & 0 \\ 0 & \ddots & & \vdots \\ & 0 \times & \vdots & 0 \\ \vdots & \times & \times & \times \\ & & 0 & \ddots \vdots \\ 0 & \dots & 0 & \times \end{bmatrix},$$

$$G^*_{u_{m,i+1}} G^*_{u_{m,(i)}} R G_{v_{m,(i)}} G_{v_{m,i+1}} = \begin{bmatrix} \times & \dots & \times & 0 \\ 0 & \ddots & & \vdots \\ & 0 \times & \vdots & 0 \\ \vdots & \mathbf{0} \times & & \times \\ & & 0 & \ddots \vdots \\ 0 & \dots & 0 & \times \end{bmatrix},$$

where

$$G_{u_{m,(i)}} = G_{u_{m,1}} G_{u_{m,2}} \dots G_{u_{m,i}},$$

and

$$G_{v_{m,(i)}} = G_{v_{m,1}} G_{v_{m,2}} \dots G_{v_{m,i}}$$

are the transformations accumulated up to the previous step. At the end of this first step we obtain :

$$G^*_{u_m} \hat{U}_2 = \begin{bmatrix} \times \dots \times & 0 \\ \vdots \ddots \vdots & \vdots \\ \times \dots \times & 0 \\ 0 \dots 0 & 1 \end{bmatrix},$$

and

$$G^*_{u_m} \hat{R} G_{v_m} = \begin{bmatrix} \times & \dots & \times & 0 \\ 0 & \ddots & & \vdots \\ & 0 \times & \vdots & 0 \\ \vdots & \mathbf{0} \times & & \times \\ & & 0 & \ddots \vdots \\ 0 & \dots & 0 & \times \end{bmatrix},$$

where

$$G_{u_m} = G_{u_{m,(n+m-1)}}, \quad \text{and} \quad G_{v_m} = G_{v_{m,(n+m-1)}}.$$

We do the same thing for the other columns of $\bar{U}_2 = G^*_{u_{j+1}} \dots G^*_{u_m} \hat{U}_2$ i.e., $\bar{U}_2(:, j)$, $j = m-1 : 1$. At the end we obtain :

$$G_{u_1}^* \dots G_{u_m}^* \hat{U}_2 = \begin{bmatrix} 0 & \dots & 0 \\ \vdots & & \vdots \\ \vdots & & \vdots \\ 0 & & \vdots \\ 1 & \ddots & \vdots \\ 0 & \ddots & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

and

$$G_{u_1}^* \dots G_{u_m}^* \hat{R} G_{v_m} \dots G_{v_1} = \begin{bmatrix} \times & \dots & \times & 0 & \dots & 0 \\ 0 & \ddots & \vdots & \vdots & & \vdots \\ & \ddots & \times & 0 & \dots & 0 \\ \vdots & & 0 & \times & \dots & \times \\ & & & \ddots & \ddots & \vdots \\ 0 & & \dots & & 0 & \times \end{bmatrix}.$$

Note that from the remark (A-1), for each iteration j we apply only $n + j - 1$ Givens rotations $G_{u_{j,i}}$. As we do k iterations, the total number of Givens rotations is

$$\sum_{j=1}^{k} n + j - 1 = \frac{k(2n + k - 1)}{2}.$$

This cost can be reduced to nk if one make a pretreatment which accelerate the procedure as it is suggested by Baker in [13]. The idea is to construct a basis $\hat{U}_2$ in the form

$$\begin{bmatrix} \times & 0 & \dots & 0 & 0 \\ \times & \times & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & 0 & \vdots \\ \times & \times & \dots & \times & 0 \\ \times & \times & \dots & \times & \times \\ & & \mathrm{X} & & \end{bmatrix},$$

before the construction of the Givens rotations G_u and G_v, we refer to [13] for more details. The construction of $\hat{U}_2$ can be obtained via a RQ decomposition.

B- Proof of Theorem 4.5

In this section we give the proof of Theorem 4.5. This result is obtained by analyzing one step i of the recursive algorithm. We first analyze the local errors in that step and hence assume all quantities at the beginning of step i to be exact. For the computations of step i we use $\bar{x}$ to denote the "computed version" of x that is actually stored in computer.

The first part of step i is the Gram-Schmidt update, which corresponds to

$$a_i = N_i, \tag{B-1}$$
$$\bar{r}_i = fl(\bar{Q}^*_{(i-1)} a_i), \tag{B-2}$$
$$\bar{\tilde{q}}_i = fl(a_i - \bar{Q}^*_{(i-1)} \bar{r}_i), \tag{B-3}$$
$$\bar{\rho}_i = fl\left(\sqrt{\bar{\tilde{q}}^*_i \bar{\tilde{q}}_i}\right), \tag{B-4}$$
$$\bar{q}_i = fl(\bar{\tilde{q}}_i / \bar{\rho}_i). \tag{B-5}$$

From (B-3), (B-5) and standard error analysis results it follows that

$$\bar{\tilde{q}}_i = a_i + d_i - \left[\bar{Q}_{(i-1)} + \delta Q_{(i-1)}\right] \bar{r}_i = \bar{\rho}_i \left[\bar{q}_i + f_i\right], \tag{B-6}$$

where (up to order ϵ_m^2) we have the element-wise inequalities

$$|[f_i]_j| \leq \epsilon_m |[\bar{q}_i]_j|, \quad |[d_i]_j| \leq n\epsilon_m |[a_i]_j|, \quad |[\delta Q_{(i-1)}]_{jl}| \leq (n-l+2)\epsilon_m |[\bar{Q}_{(i-1)}]_{jl}|.$$

To obtain this result we assumed that the loop on the columns of the Gram-Schmidt orthogonalization (B-3) progresses from left to right. We can then equate this as follows :

$$a_i + e_i = \left[\bar{Q}_{(i-1)} \middle| \bar{q}_i\right] \begin{bmatrix} \bar{r}_i \\ \rho_i \end{bmatrix}, \quad e_i = d_i - \delta Q_{(i-1)} \bar{r}_i + f_i \bar{\rho}_i. \tag{B-7}$$

We also assume that

$$\| \left[\bar{Q}_{(i-1)} \middle| \bar{q}_i\right] - \left[Q_{(i-1)} \middle| q_i\right] \|_2 = K.\epsilon_m \ll 1, \tag{B-8}$$

i.e., there is no *complete* loss of orthogonality, which allows us to approximate the 2-norm of $\left[\bar{Q}_{(i-1)} \middle| \bar{q}_i\right]$ or any of its columns by $1 + O(\epsilon_m)$. We then obtain the inequalities :

$$\begin{aligned}
\|e_i\|_2 &\leq \|d_i\|_2 + \|f_i \bar{\rho}_i\|_2 + \sum_l \| |\delta Q_{(i-1)}|_{:,l} \|_2 . |\bar{r}_i|_l + O(\epsilon_m^2) \\
&\leq \epsilon_m \left[n\|a_i\|_2 + \|\bar{q}_i\| \bar{\rho}_i + \sum_l \| |Q_{(i-1)}|_{:,l} \|_2 . (n-l+2) |r_i|_l \right] + O(\epsilon_m^2) \\
&\leq \epsilon_m \left[n\|a_i\|_2 + (|\bar{\rho}_i| + \sum_l (n-l+2) |\bar{r}_i|_l) \right] + O(\epsilon_m^2) \\
&\leq \epsilon_m \left(n\|a_i\|_2 + \|[1, 2, \ldots, n+1]\|_2 \|a_i\|_2 \right) + O(\epsilon_m^2) \\
&\leq \epsilon_m \left(n + \sqrt{\frac{(n+2)^3}{3}} \right) \|a_i\|_2 + O(\epsilon_m^2)
\end{aligned} \tag{B-9}$$

where the next-to-last line was obtained by Cauchy Schwarz. Notice that all errors due to this part are superposed on column a_i. Therefore the error matrix E_1 of this first part satisfies $\|E_1\|_F = \|e_i\|_2$.

The second part of step i consists of the transformations G_v and G_u in (4.8), which we assume are each implemented with a sequence of n Givens rotations. For this we will use Lemma 18.8 of [69], which we recall in a slightly modified form. We refer to §.1.1.2, Appendix 8 and [69] for the details of the implementation and construction of each Givens rotation.

Lemma 8.1.

Consider the sequence of Givens transformations

$$L_n = G_n \ldots G_1 L = G \cdot L.$$

Then there exists a perturbation ΔL of L so that the computed matrix $\bar{L}_n$ satisfies

$$\bar{L}_n = G(L + \Delta L), \quad \|\Delta L\|_F \leq 6n\sqrt{2}\epsilon_m \|L\|_F + O(\epsilon_m^2).$$

■

Applying this to the products $Q_{up} \cdot R_{up} = (QG_u^*) \cdot (G_u R G_v^*)$ and $V_{up} = (VG_v^*)$ we obtain

$$\begin{aligned} \bar{Q}_{up}\bar{R}_{up} &= (Q + \Delta Q)G_u^* \cdot G_u(R + \Delta R)G_v^* \doteq QRG_v^* + E_2, \\ \bar{V}_{up} &= (V + \Delta V)G_v^* \doteq VG_v^* + F, \end{aligned}$$

where

$$\begin{aligned} E_2 &\doteq (\Delta QR + Q\Delta R + \Delta Q\Delta R)G_v^*, \\ \|\Delta Q\|_F &\leq 6\sqrt{2}n\epsilon_m\|Q\|_F + O(\epsilon_m^2) = 6n\sqrt{2(n+1)}\epsilon_m + O(\epsilon_m^2), \\ \|\Delta R\|_F &\leq 12\sqrt{2}n\epsilon_m\|R\|_F + O(\epsilon_m^2) = 12n\sqrt{2(n+1)}\epsilon_m\|M\|_2 + O(\epsilon_m^2), \end{aligned}$$

and

$$\begin{aligned} F &\doteq (\Delta V)G_v^*, \\ \|\Delta V\|_F &\leq 6\sqrt{2}n\epsilon_m\|V\|_F + O(\epsilon_m^2) = 6n\sqrt{2(n+1)}\epsilon_m + O(\epsilon_m^2). \end{aligned}$$

The norms of E_2 and F can then be bounded by :

$$\begin{aligned} \|E_2\|_F &\leq \|Q\|_2\|\Delta R\|_F + \|R\|_2\|\Delta Q\|_F + O(\epsilon_m^2) \\ &\leq 18n\sqrt{2(n+1)}\epsilon_m\|M\|_2 + O(\epsilon_m^2), \\ \|F\|_F &\leq 6n\sqrt{2(n+1)}\epsilon_m + O(\epsilon_m^2). \end{aligned}$$

Combining the bounds for E_1 and E_2 yields the bound

$$\|E\|_F \ \leq \ 26\epsilon_m n^{\frac{3}{2}}\|M\|_2 + O(\epsilon_m^2)$$

for the local error E in step i. Similarly, the error matrix F on $V_{(i)}$ corresponding to the local errors of step i can be bounded by

$$\|F\|_F \ \leq \ 9\epsilon_m n^{\frac{3}{2}} + O(\epsilon_m^2).$$

In order to sum up these errors over the τ steps of the algorithm, we can neglect the second order effects and then only need to multiply these bounds by $(m - n)$. This then yields the bounds of Theorem 4.5.

References

1. *The control handbook*. Editor : W.S. Levine. CRC Press and IEEE Press, 1996.
2. Abdelmalek, N. Round-off error analysis for gram-schmidt method and solution of linear least squares problems. *BIT 11* (1971), 45–68.
3. Adamjan, V., Arov, D. Z., and Krein, M. G. Infinite block hankel matrices and related extensions problems. *AMS Transl. 111* (1978), 133–156.
4. Anderson, B., and Skelton, R. The generation of all q-markov covers. *IEEE Trans. Circuits Syst 35*, 4 (1988), 375–384.
5. Antoulas, A. *Lectures on the approximation of large-scale dynamical systems*. Siam book series : Advances in design and control, 2002.
6. Antoulas, A., Sorenson, D., and Gugercin, S. A survey of model reduction methods for large-scale systems. *AMS-IMS-SIAM Summer Research Conference on Structured Matrices, Boulder* (1999).
7. Antoulas, A., Sorenson, D., and Zhou, Y. On the decay rate of Hankel singular values and related issues. *CAAM Technical Report TR01-09, Rice University* (2001), Available from http://www.caam.rice.edu/caam/caam–techrep.html.
8. Arnold, V. I. *Mathematical methods of classical mechanics*. Springer Verlag, 1978.
9. Arnoldi, W. The principle of minimized iterations in the solution of the matrix eigenvalues problem. *Quart. Apppl. Math 9* (1951), 17–29.
10. Åström, K., and Wittenmark, B. *Computer-controlled systems. Theory and design*. Prentice Hall, 1997.
11. Axelson, O. *Iterative solution methods*. Cambridge University Press, Cambridge, 1994.
12. Bai, Z., Dewilde, P. M., and Freund, R. W. Reduced-order modeling. *Numerical Analysis Manuscript No.02-4-13, Bell Laboratories, Murray Hill, New jersey* (March 2002), Available from http://cm.bell–labs.com/cs/doc/02.
13. Baker, C. An incremental block algorithm for tracking dominant subspaces. *Technical Report FSU-CSIT-03-03, CSIT, The Florida State University* (2003).
14. Barrett, R., Berry, M., Chan, T., Demmel, J., Donato, J., Dongarra, J., Eijkhout, V., Pozo, R., Romine, C., and der Vorst, H. V. *Templates for the solution of linear systems : building blocks for iterative methods*. SIAM Publications, Philadelphia, PA, 1994.
15. Bartels, R., and Stewart, G. Algorithm 432 : solution of the matrix equation $AX + XB = C$. *Commun. ACM 15* (1972), 820–826.
16. Beck, C., Doyle, J., and Glover, K. Model reduction of multidimensional and uncertain systems. *IEEE Trans. Automat. Control 41* (1996), 1466–1477.
17. Benner, P., Castillo, M., Quintana-Ortí, E., and Hernández, V. Parallel partial stabilizing algorithms for large linear control systems. *J. Supercomput. 615*, 2 (2000), 193–206.
18. Benner, P., Quintana-Ortí, E., and Quintana-Ortí, G. Balanced truncation model reduction of large-scale dense systems on parallel computers. *Math. Comput. Model. Dyn. Syst. 6*, 4 (2000), 383–405.
19. Benner, P., Quintana-Ortí, E., and Quintana-Ortí, G. Numerical solution of discrete stable linear matrix equations on multicomputers. *Parallel Algorithms Appl. 17*, 2 (2002), 127–146.
20. Berry, M. *Computational information retrieval*. SIAM Publications, 2001.
21. Berry, M., DRmač, Z., and Jessup, E. Matrices, vector spaces, and information retrieval. *Siam Review 41* (1999), 335–362.
22. Birkhoff, G., Varga, R., and Young, D. Alternating direction implicit methods. *In advances in Computers, Academic Press, New York 3* (1962), 189–273.

23. Björk, A. Solving linear least squares problems by gram-schmidt orthogonalization. *BIT 7* (1967), 1–21.
24. Björk, A. Numerics of gram-schmidt orthogonalization. *Lin. Alg. and Its Applic. 197/198* (1994), 297–316.
25. Board, J., and Schulten, K. The fast multipole algorithm. *IEEE, Computing in Science and Engineering 2*, 1 (2000), 76–79.
26. Boley, D. Krylov space methods on state-space control models. *Circuits Systems Signal Process 13(6)* (1994), 733–758.
27. by the staff of Engineering Research Associates, Inc. *High-speed computing devices.* McGraw-Hill, 1950.
28. Callier, F., and Desoer, C. *Linear system theory.* Springer-Verlag, 1991.
29. Chahlaoui, Y., Gallivan, K., and Van Dooren, P. An incermental method for computing dominant singular subspaces. *in Computational Information Retrieval, ed. Michael W. Berry, SIAM Publications* (2001), 53–62.
30. Chahlaoui, Y., Gallivan, K., and Van Dooren, P. Recursive calculation of dominant singular subspaces. *SIAM Journal on Matrix Analysis and Applications 25*, 2 (2003), 445–463.
31. Chahlaoui, Y., Lemonnier, D., Meerbergen, K., Vandendorpe, A., and Van Dooren, P. Model reduction of second order systems. *MTNS2002 Fifteenth International Symposium on Mathematical Theory of Networks and Systems, University of Notre Dame, August 12-16* (2002).
32. Chahlaoui, Y., Lemonnier, D., Vandendorpe, A., and Van Dooren, P. Second order balanced truncation. *submitted to Linear Algebra Appl., Special Issue on Model Reduction* (2003).
33. Chahlaoui, Y., and Van Dooren, P. A collection of benchmark examples for model reduction of linear time invariant dynamical systems. *SLICOT Working Note 2002-2* (2002), Available from ftp://wgs.esat.kuleuven.ac.be/pub/WGS/REPORTS/SLWN2002–2.ps.Z.
34. Chan, T. An improved algorithm for computing the singular value decomposition. *ACM Trans. Math. Soft 8* (1982), 72–83.
35. Chandrasekaran, S., and Gu, M. A divide-and-conquer algorithm for the eigendecomposition of symmetric block-diagonal plus semiseparable matrices. *accepted for publication in Numerische Mathematik* (1999).
36. Chandrasekaran, S., and Gu, M. Fast and stable algorithms for banded plus semi-separable matrices. *submitted to SIAM J. Matrix Ana. Appl* (2000).
37. Chandrasekaran, S., and Gu, M. Fast and stable eigendecomposition of symmetric banded plus semi-separable matricesalgorithms for banded plus semi-separable matrices. *Linear Algebra and its applications 313*, 1–3 (2000), 107–114.
38. Chandrasekaran, S., and Gu, M. A fast and stable solver for recursively semi-separable systems of equations. *Structured matrices in mathematics, computer science and engineering II, edited by Vadim Olshevsky, in the Contemporary Mathematics series, AMS publications* (2001).
39. Chandrasekaran, S., Gu, M., and Pals, T. A fast ans stable solver for smooth recursively semi-separable systems. *Paper presented at the SIAM Annual Conference, San Diego, CA, 2001, and SIAM Conference of Linear Algebra in Controls, Signals and Systems, Boston, MA* (2001).
40. Chandrasekaran, S., Manjunath, B., Wang, Y., Winkeler, J., and Zhang, H. An eigenspace update algorithm for image analysis. *Graphical Models and Image Processing 59*, 5 (1997), 321–32.
41. Chang, X., Paige, C., and Stewart, G. Perturbation analyses for the QR factorization. *SIAM J. Matr. Anal. Appl 18* (1997), 775–791.
42. Chatelin, F. *Spectral approximation of linear operators.* Academic Press, 1983.
43. Chiprout, E., and Nakhla, M. Generalized moment-matching methods for transient analysis of interconnect networks. *Proc. of the 29th ACM/IEEE Design Automation Conference* (1992), 201–206.
44. Chu, K. W. E. The solution of the matrix equation $AXB - CXD = Y$ and $(YA - DZ, YC - BZ) = (E, F)$. *Linear Algebra Appl. 93* (1987), 93–105.
45. Ciarlet, P. *Introduction to numerical linear algebra and optimisation.* Cambridge U. Press, Cambridge, UK, 1989.
46. de Villemagne, C., and Skelton, R. Model reductions using a projection formulation. *Int. J. Control 46* (1987), 2141–2169.
47. Demmel, J. *Applied numerical linear algebra.* SIAM Publications, Philadelphia, 1997.
48. Dewilde, P., and van der Veen, A.-J. *Time-varying systems and computations.* Kluwer Academic Publishers, Boston, MA, 1998.
49. Elfadel, I., and Ling, D. A block rational arnoldi algorithm for multipoint passive model-order reduction of multiport RLC networks. *Proc. of the International Conference on Computer-Aided Design* (1997), 66–71.
50. Enns, D. Model reduction with balanced realizations : An error bound and frequency weighted generalization. *Proc. of the IEEE Conference on Decision and Control* (1981), 127–132.

51. Feldmann, P., and Freund, R. Efficient linear circuit analysis by padé approximation via the lanczos process. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 14(5)* (1995), 639–649.
52. Freund, R., and Nachtigal, N. Qmr : A quasi-minimal residual method for non-hermitian linear systems. *Numer. Math. 60* (1991), 315–339.
53. G.A. Baker, J., and Graves-Morris, P. Padé approximations part 1 and 2. *Encyclopedia of mathematics and its applications, Addison-Wesley* (1981).
54. Gallivan, K., Grimme, E., and Van Dooren, P. A rational lanczos algorithm for model reduction. *Numerical Algorithms 12* (1996), 33–63.
55. Gallivan, K., Grimme, E., and Van Dooren, P. Model reduction of large-scale systems rational krylov versus balancing techniques. *Error Control and Adaptivity in Scientific Computing* (1999), 177–190.
56. Gallivan, K., Vandendorpe, A., and Van Dooren, P. Sylvester equations and projection-based model reduction. *J. Comp. Appl. Math. Special Issues* (2002), accepted.
57. Gallivan, K., Vandendorpe, A., and Van Dooren, P. Model reduction via truncation : an interpolation point of view. *Linear Algebra Appl.* (2003), submitted.
58. Gardiner, J., Laub, A. J., Amato, J. J., and Moler, C. B. Solution of the sylvester matrix equation $AXB^T + CXD^T = E$. *ACM Trans. Math. Software 18* (1992), 223–231.
59. Gawronski, W. K. *Dynamics and control of structures. A modal approach.* Mechanical engineering series, Springer, 1998.
60. Geman, S. A limit theorem for the norm of random matrices. *Annals of Probability 8* (1980), 252–261.
61. Glover, K. All optimal Hankel norm approximations of linear multivariable systems and their $\mathcal{L}^\infty$-error bounds. *Internat. J. Control 39* (1984), 1115–1193.
62. Golub, G., Nash, S., and Loan, C. V. A hessenberg-schur method for the matrix problem $AX + XB = C$. *IEEE Trans. Autom. Contr. AC-24(6)* (1979), 909–913.
63. Golub, G., and Van Loan, C. *Matrix Computations.* Johns Hopkins University Press, Baltimore MD, 1996.
64. Grimme, E. Krylov projection methods for model reduction. *PhD thesis, University of Illinois at Urbana-Champaign* (1997).
65. Grimme, E., Sorenson, D., and Van Dooren, P. Model reduction of state space systems via an implicitly restarted lanczos method. *Numer. Algorithms 12(1-2)* (1996), 1–31.
66. Gugercin, S., Antoulas, A., and Bedrossian, N. Approximation of the international space station 1r and 12a flex models. *Proceedings of the 40th IEEE Conference on Decision and Control* (2001).
67. Gugercin, S., Sorenson, D., and Antoulas, A. A modified low-rank Smith method for large-scale Lyapunov equations. *Numerical Algorithms 32 (1)*, 27–55.
68. Hammarling, S. J. Numerical solution of the stable, non-negative definite lyapunov equation. *In R.V. Patel, A.J. Laub, and P.M. Van Dooren, editors, Numerical Linear Algebra Techniques for Systems and Control, IEEE Press, New York, NY, USA* (1994), 500–516.
69. Higham, N. *Accuracy and Stability of Numerical Algorithms.* SIAM Publications, Philadelphia, 1996.
70. Horn, R., and Johnson, C. *Topics in matrix analysis.* Cambridge U.press, Cambridge, UK, 1991.
71. Hu, D., and Reichel, L. Krylov-subspace methods for the sylvester equation. *Linear Algebra Appl 172* (1992), 283–313.
72. Hyland, D., and Bernstein, D. The optimal projection equations for model reduction and the relationships among the methods of wilson, skelton and moore. *IEEE Trans. Automat. Contr. AC-30. 12* (1985), 1201–1211.
73. Imae, J., Perkins, J., and Moore, J. Toward time-varying balanced realization via Riccati equations. *Math. Control Signals Systems 5* (1992), 313–326.
74. Jaimoukha, I., and E.M.Kasenally. Krylov subspace methods for solving large lyapunov equations. *SIAM J. Numer. Anal. 31(1)* (1994), 227–251.
75. Jaimoukha, I., and E.M.Kasenally. Implicitly restarted krylov subspace methods for stable partial realizations. *SIAM J. Matrix Anal. Appl. 18(3)* (1997), 633–652.
76. Jbilou, K. Numerical approximate solutions to large stein equations. *TR, Pub. 155 LMPA–Universit du littoral* (2001).
77. Jonsson, I., and Kågström, B. Recursive blocked algorithms for solving triangular matrix equations – part ii : Two-sided and generalized sylvester and lyapunov equations. *SLICOT Working Note 2001–5*.
78. Jr, W. W., and Johnston, P. R. *Structural dynamics by finite elements.* Prentice Hall, Inc, 1987.
79. Kågström, B., and Poromaa, P. LAPACK-style algorithms and software for solving the generalized sylvester equation and estimating the separation between regular matrix pairs. *ACM Trans. Math. Software 22(1)* (1996), 78–103.
80. Kailath, T. *Linear systems.* Prentice-Hall, 1980.

81. Kamon, M., Wang, F., and White, J. Generating nearly optimally compact models from krylov-subspace based reduced-order models. *IEEE Transactions on Circuits and Systems II : Analog and Digital Signal Processing 47(4)* (2000), 239–248.
82. Lall, S., Krysl, P., and Marsden, J. E. Structure-preserving model reduction of mechanical systems. *Submitted to Dynamics and Stability of Systems* (2000).
83. Lanczos, C. An iteration method for the solution of the eigenvalue problem linear differential and integral operators. *J. Res. Natl. Bur. Stand 45* (1950), 255–282.
84. Li, J.-R. *Model reduction of large linear systems via low rank system Gramians.* PhD thesis, Mathematics, MIT, 2000.
85. Li, J.-R., Wang, F., and White, J. An efficient lyapunov equation-based approach for generating reduced-order models of interconnect. *Proc. of the 36th Design Automation Conference* (1999), 1–6.
86. Liu, K., and Skelton, R. A new formulation of q-markov covariance equivalent realization. *Appl. Math. Comput 53*, 1 (1993), 83–95.
87. Liu, K., and Skelton, R. q-markov covariance equivalent realization and its application to flexible structure identification. *J. Guid. Control Dyn 16*, 2 (1993), 308–319.
88. Lu, A., and E.L.Wachspress. Solution of lyapunov equations by alternating direction implicit iteration. *Comput. Math. Appl. 21(9)* (1991), 43–58.
89. Meirovitch, L. *Dynamics and control of structures.* Wiley, New York, 1990.
90. Meyer, D. G., and Srinivasan, S. Balancing and model reduction for second-order form linar systems. *IEEE Trans. Automat. Control 41*, 11.
91. Moore, B. Principal component analysis in linear systems : controllability, observability, and model reduction. *IEEE Trans. Automat. Control 26* (1981), 17–31.
92. Obinata, G., and Anderson, B. D. O. *Model Reduction for Control System Design.* Springer-Verlag, London, 2001.
93. Ogata, K. *Discrete-time control systems.* Prentice Hall, 1987.
94. Ortega, J. M., and Rheinboldt, W. C. *Iterative Solution of Nonlinear Equations in Several Variables.* Academic Press, San Diego, 1970.
95. Penzl, T. Numerical solution of generalized lyapunov equations. *Advances in Comp. Math 8* (1998), 33–48.
96. Penzl, T. Algorithms for model reduction of large dynamical systems. *Technical Report SFB393/99-40, Sonderforschungsbereich 393 Numerische Simulation auf massiv parallelen Rechnern TU Chemnitz, 09107 Chemnitz, FRG* (1999), Available from http://www.tu–chemnitz.de/sfb393/sfb99pr.html.
97. Penzl, T. A cyclic low-rank Smith method for large sparse Lyapunov equations. *Siam J. Sci. Comput. 21(4)* (2000), 1404–1418.
98. Penzl, T. Eigenvalue decay bounds for solutions of lyapunov equations : the symmetric case. *Systems and Control Letters* (2000).
99. Pernebo, L., and Silverman, L. Model reduction via balanced state space representations. *IEEE Trans. Automat. Control 27(2)* (1982), 382–387.
100. Rabiei, P., and Pedram, M. Model order reduction of large circuits using balanced truncation. *Proc. of the Design Automation Conference, Asia and South Pacific 1* (1999), 237–240.
101. Rubinstein, M. *Structural systems-statics, dynamics and stability.* Prentice-Hall, Inc, 1970.
102. Ruhe, A., and Skoogh, D. Rational krylov algorithms for eigenvalue computation and model reduction. *Applied parallel computing, Springer, Berlin* (1998), 491–502.
103. Saad, Y. *Iterative methods for sparse linear systems.* PWS Publishing Compagny, Boston, 1996.
104. Saad, Y., and Schultz, M. Gmres : A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Sci. Stat. Comput. 7(3)* (1986), 856–869.
105. Safonov, M., and Chiang, R. A schur method for balanced-truncation model reduction. *IEEE Trans. Automat. Control 34(7)* (1989), 729–733.
106. Sandberg, H., and Rantzer, H. Balanced model reduction of linear time-varying systems. *IFAC02, 15th Triennial World Congress, Barcelona, Spain* (2002).
107. Shokoohi, S., Silverman, L., and Van Dooren, P. Linear time-variable systems : Balancing and model reduction. *IEEE Trans. Automat. Control 28* (1983), 810–822.
108. Sima, V., Benner, P., Huffel, S. V., and Varga, A. Improving the efficiency and accuracy of the MATLAB control toolbox using SLICOT-based gateways. *Proceeding of the MATHEMATICAL THEORY OF NETWORKS AND SYSTEMS MTNS 98, Padova, Italy* (1998).
109. Skelton, R., and Anderson, B. q-markov covariance equivalent realizations. *Int. J. Control 44* (1986), 1477–1490.

110. Skelton, R., and Anderson, B. Weighted q-markov covariance equivalent realizations. *Int. J. Control 49*, 5 (1989), 1755–1771.
111. Skoogh, D. Krylov subspace methods for linear systems, eigenvalues and model order reduction. *PhD thesis, Chalmers University of Technology* (1998).
112. Soize, C. Reduced models in the medium frequency range for general dissipative structural dynamics systems. *Eur. J. Mechan. and Solids 17* (1998), 657–685.
113. Sreedhar, J., and Van Dooren, P. A Schur approach for solving some periodic matrix equations. Systems and Networks : Mathematical theory and applications. *Mathematical Research 77* (1994), 339–362.
114. Stewart, G. On the early history of the singular value decomposition. *SIAM Review 35* (1993), 551–566.
115. Stewart, G. *Matrix algorithms*, vol. 1 & 2. SIAM Publications, Philadelphia, 2001.
116. Stewart, G., and Sun, J. *Matrix Perturbation Theory*. Academic Press, San Diego, 1990.
117. Su, T. J., and Craig, J. R. R. Krylov model reduction algorithm for undamped structural dynamics systems. *J. Guidance Control Dynamics 14*, 6.
118. Su, T. J., and Craig, J. R. R. Model reduction and control of flexible structures using krylov vectors. *J. Guidance Control Dynamics 14*, 2.
119. Su, T. J., and Craig, J. R. R. An unsymmetric lanczos algorithm for damped structural dynamics systems. *Proceeding, 33rd Conference on Structures, Structural Dynamics and Materials. AIAA/ASME/ASCE/AHS/ASC*.
120. Tisseur, F., and Meerbergen, K. The quadratic eigenvalue problem. *Siam Review 43*, 2 (2001), 235–286.
121. Tornero, J., Albertos, P., and Salt, J. Periodic optimal control of multirate sampled data systems. *IFAC. Periodic Control Systems. PSYCO2001. Italy* (2001), 199–204.
122. Trefethen, L., and III, D. B. *Numerical linear algebra*. SIAM Publications, Philadelphia, 1997.
123. Tsakalis, K., and Ioannou, P. Linear time-varying systems : Control and adaptation. *Prentice Hall* (1993).
124. Tufts, D., and Kunaresan, R. Singular value decomposition and improved frequency estimation using linear prediction. *IEEE Trans.Acoustics, Speech, and Signal processing ASSP-30* (1982), 671–675.
125. van der Veen, A.-J., and Dewilde, P. On low-complexity approximation of matrices. *Linear Algebra and its Applications*, 205–206, 1145–1202.
126. Van Dooren, P. Orthogonal matrix decompositions in systems and control. *Error Control and Adaptivity in Scientific Computing, Eds. Bulgak and Zenger C-536* (1999), 159–175.
127. Van Dooren, P. Software for control systems analysis and design, singular value decomposition. *Encyclopedia of Electrical and Electronics Engineering* (1999), 464–473.
128. Van Dooren, P. Gramian based model reduction of large-scale dynamical systems. *in Numerical Analysis 1999* (2000), 231–247.
129. Vandendorpe, A., and Van Dooren, P. Projection of state space realizations. *Open problems in Mathematical Systems and Control Theory Eds. R. Brockett et al.* (2003).
130. Varga, A. Balancing related methods for minimal realization of periodic systems. *Systems & Control Letters 36* (1999), 339–349.
131. Varga, A., and Van Dooren, P. Computational methods for periodic systems - an overview. *Prepr. IFAC Workshop on Periodic Control Systems, Como, Italy* (2001), 171–176.
132. Verlaan, M., and Heemink, A. Tidal flow forecasting using reduced rank square root filters. *Stochastic Hydrology and Hydraulics 11* (1997), 349–368.
133. Verriest, E., and Kailath, T. On generalized balanced realizations. *IEEE Trans. Automat. Control 28(8)* (1983), 833–844.
134. Wang, W., and Safonov, M. Relative error bound for discrete balanced truncation. *International J. of Control 54* (1991), 593–612.
135. Wilkinson, J. *The algebraic eigenvalue problem*. Oxford University Press, New York, 1965.
136. Wortelboer, P. Frequency-weighted balanced reduction of closed-loop mechanical servo-systems : theory and tools. *PhD thesis, Delft University of Technology* (1994).
137. Zhou, K., Doyle, J., and Glover, K. *Robust and optimal control*. Prentice Hall, 1995.

Université catholique de Louvain
Faculté des sciences appliquées
Département d'ingénierie mathématique (INMA)
Center for Systems Engineering and Applied Mechanics (CESAME)

Avenue Georges Lemaître 4, 1348 Louvain-la-Neuve, Belgique
Tél. 32 (0) 10 47 25 97 – fax 32 (0) 10 47 21 80
E-mail chahlaoui@auto.ucl.ac.be
http://www.csam.ucl.ac.be

The basic idea of model reduction is to represent a complex linear dynamical system by a much simpler one. This may refer to many different techniques, but in this dissertation we focus on projection-based model reduction of linear systems. The projection is based on the dominant eigen-spaces of energy functions for ingoing and outgoing signals of the system.

These energy functions are called Gramians of the system and can be obtained as the solutions of Stein equations. When the system matrices are large and sparse, it is not obvious how to compute efficiently these solutions or their dominant eigen-spaces. In fact, direct methods ignore sparsity in the Stein equations and are not very attractive for parallelization. Their use is then limited if the state dimension N of the system is large. The complexity of these methods is roughly $O(N^3)$ floating point operations and they require about $O(N^2)$ words of memory.

This thesis provides some new ideas of recursive projection-based model reduction for time-varying systems as well as time-invariant systems. We present three algorithms for the recursive computation of the projection. These algorithms combine ideas of two classical methods — namely Balanced Truncation and Krylov subspaces — to produce a low-rank approximation of the Gramians or the input/output map of the system. We show the practical relevance of our results with real world benchmark examples. We also present some new ideas for second order systems. Such systems have a special structure which one wants to preserve in the reduced order model. We show how to adapt our projection based method to such systems.

Younès Chahlaoui was born in El Jadida (Morocco) on November 19, 1974.
He first obtained a Baccalauréat in Mathematical Sciences from the Academy of El Jadida, in 1993. He then joined the Faculty of Sciences at the Université Chouaïb Doukkali in El Jadida, where he obtained, in 1999, a Bachelor Degree in Applied Mathematics (Licence + DEUG).
He finally obtained a Master Degree (DEA), and a Doctoral Degree in Applied Mathematics, both at the Université catholique de Louvain (Belgium) in 2001 and 2003, respectively.
He held research and teaching positions at the Université catholique de Louvain from 1999 to 2003.
Research interests: Numerical Analysis (PDE's, ODE's), Scientific Computing, Systems and Control Theory, Numerical Simulation.

IAP V/22 systems & control

CESAME

UCL

Younès Chahlaoui

Low-rank approximation and model reduction

2003