Universidade Federal de Minas Gerais
Université Grenoble Alpes

Building Reduced Order Models for Dynamical Simulation and Optimization of Numerical Electromagnetic Models

Mateus Antunes Oliveira Leite

Thesis submitted to the Graduate Program in Electrical Engineering of the Federal University of Minas Gerais (PPGEE) and to the Graduate Program of Grenoble Alpes University (EEATS) in partial fulfillment of the requirements for obtaining the degree of PhD in Electrical Engineering.

Advisor: Prof. Dr. João Antônio de Vasconcelos
Advisor: Prof. Dr. Benoit Delinchant
Co-Advisor: Prof. Dr. Jean-Michel Guichon

Belo Horizonte, 22th February 2018

Communauté
UNIVERSITÉ Grenoble Alpes

UFMG UNIVERSIDADE FEDERAL DE MINAS GERAIS

THÈSE

Pour obtenir le grade de

DOCTEUR DE LA COMMUNAUTE UNIVERSITE GRENOBLE ALPES

préparée dans le cadre d'une co-tutelle *entre la Communauté Université Grenoble Alpes* et l'Université Fédérale du Minas Gerais

Spécialité : GENIE ELECTRIQUE

Arrêté ministériel : le 6 janvier 2005 – 25 mai 2016

Présentée par

Mateus ANTUNES OLIVEIRA LEITE

Thèse dirigée par **Benoit DELINCHANT** et **João ANTÔNIO DE VASCONCELOS**
codirigée par **Jean-Michel GUICHON**

préparée au sein des **Laboratoires G2ELAB et Laboratoire de Calcul Évolutionnaire**

dans **les Écoles Doctorales EEATS et PPGEE**

Construction de modèles réduits pour la simulation dynamique et optimisation de modèles électromagnétiques numériques

Thèse soutenue publiquement le **24 avril 2018**,
devant le jury composé de :

M. Benoit DELINCHANT
Maître de conférence à l'Université de Grenoble Alpes (Membre)
M. João ANTÔNIO DE VASCONCELOS
Professeur à l'UFMG (Membre)
M. Jean-Michel GUICHON
Maître de conférence à l'Université de Grenoble Alpes (Membre)
M. Sthépane CLENET
Professeur à l'École Nationale Supérieure d'Arts et Métiers (Rapporteur)
M. Adroaldo RAIZER
Professeur à l'UFSC (Président du jury)
M. Marcio MATIAS AFONSO
Professeur Associé à CEFET-MG (Membre)
M. Ricardo LUIZ DA SILVA ADRIANO
Professeur Associé à l'UFMG (Invité)

Contents

Abstract **iv**

Résumé **v**

List of Figures **viii**

List of Tables **ix**

List of Symbols **x**

List of Acronyms **xi**

1 Introduction **1**
1.1 Motivation to reach very fast computer models 1
1.1.1 Simulation of Electromagnetic Devices 3
1.1.2 Optimization of Electromagnetic Devices 5
1.2 Different Model Order Reduction Approaches 9
1.2.1 Vector Fitting 9
1.2.2 Balanced Truncation 10
1.2.3 Proper Orthogonal Decomposition 10
1.2.4 Moment Matching 11
1.3 Objectives 11
1.4 Structure of the Thesis 13
1.5 A Word on Notation 14
1.6 Conclusion 15

2 Constructing Reduced Order Models **16**
2.1 Introduction 16
2.1.1 The State Space Representation 16
2.1.2 Model Order Reduction by Projection 20
2.2 Techniques to Building Reduced Order Models 22
2.2.1 Balanced Truncation 23
2.2.2 Proper Orthogonal Decomposition 30
2.2.3 Moment Matching 36
2.2.4 Circuits and special cases 49
2.3 Conclusion 50

3 Improving Moment Matching: Accuracy, Stabilization and Time Simulation 51
3.1 Introduction . . . 51
3.2 Placing Many Expansion Points . . . 52
3.2.1 Block Arnoldi Process . . . 52
3.3 Real Subspace for Time Domain Simulation . . . 55
3.4 Stability of Reduced Order Models . . . 58
3.4.1 Rank-1 Updates . . . 59
3.4.2 Rank-2 Updates . . . 62
3.5 Adaptive Placement of Expansion Points . . . 67
3.5.1 Adaptive Algorithm . . . 67
3.5.2 Greedy Algorithm . . . 72
3.6 Example of Application . . . 74
3.7 Conclusion . . . 78

4 Exploiting Reduced Order Models for Simulation 79
4.1 Introduction . . . 79
4.2 Laminated Bus Bar Model . . . 80
4.2.1 Application . . . 84
4.3 Electromagnetic Wave Scattering Problem . . . 86
4.3.1 Application . . . 92
4.4 Conclusion . . . 94

5 Optimization over Reduced Order Models 95
5.1 Introduction . . . 95
5.2 General Purpose Interpolation Methods . . . 96
5.2.1 Radial Basis Interpolation . . . 97
5.2.2 Kriging . . . 99
5.3 Interpolation of Reduced Order Models . . . 105
5.3.1 Subspace Interpolation . . . 105
5.3.2 Matrix Interpolation . . . 110
5.4 Conclusion . . . 114

6 Application of Model Order Reduction in Optimization 115
6.1 Introduction . . . 115
6.2 Optimization Algorithm . . . 115
6.2.1 Optimization with Surrogate Models . . . 116
6.3 The Problem of Placing Capacitors at a Power Bus . . . 119
6.4 The Simplex Sampling Strategy . . . 122
6.5 The Problem of Placing Wires in a Cage . . . 127
6.6 Conclusion . . . 130

7 Conclusion 131

Appendices 133

A Updates of the Generalized Eigenvalue Problem 133

A.1 Rank-1 Update of the Generalized Eigenvalue Problem 133
A.2 Rank-N Update of the Generalized Eigenvalue Problem 137
A.3 Real Rank-2 Updates 139

B Subspace produced by the modified Arnoldi Iteration **141**
B.1 Statement of the problem 141
B.1.1 Mathematical background 143
B.2 Proof of Equivalence of Subspaces 146

C Identities Involving Vectors **150**
C.1 Introduction 150
C.2 Derivatives 150
C.2.1 Validity of the covariance kernel 152

D Electric Circuit Equations **155**

E Kriging **159**
E.1 Introduction 159
E.2 No Bias Condition 159
E.3 The variance identity 160
E.4 Variance of the Error 160
E.5 Solving the restricted problem 161
E.6 Maximum Likelihood Estimation 161
E.7 Explicit solution for the Kriging linear system 164

Bibliography **175**

Abstract

There are many available methods capable of producing High-Fidelity Models (HFM) of electromagnetic systems. If the required precision is very high or the nature of the phenomenon that is being modeled is complex, a very large number of equations may have to be solved. If the model is used for applications where many different geometries or parameters must be considered, as is the case in design optimization, solving these equations many times can be very time consuming.

To avoid the burden of this computation, Model Order Reduction (MOR) algorithms have been developed. They consist in procedures of finding Reduced Order Models (ROMs) that accurately describe the input/output behavior of the High-Fidelity Model but using only a very small number of equations.

In this work, MORs techniques are analyzed and improved. Special attention is paid to Moment Matching. Problems like placement of expansion points, stability and numerical robustness are investigated. This has allowed the simulation and optimization of complex electromagnetic device. As examples of application we have presented a laminates bus bar and a wave scattering problem.

In addition to that, a smart adaptive way of sampling the design space to allow fast optimization has been developed. The sampled points are used to perform interpolation and approximate the objective function in a very fast manner. Classic optimization techniques have also been coupled with the Reduced Order Models, accelerating the computations. The proposed approaches have been tested mainly in electromagnetic problems obtained by the Partial Equivalent Circuit Element (PEEC) method.

Résumé

Il y a des nombreuses méthodes capables de produire des modèles numériques très précis des systèmes électromagnétiques. Si la précision demandée est très importante ou la nature du phénomène modélisé est très complexe, il faudra résoudre un nombre élevé d'équations. Si le modèle est utilisé dans des applications où de nombreux paramètres doit être prises en compte, comme c'est le cas de la conception optimale, la résolution de ces équations est souvent très couteuse.

Pour éviter ce calcul, des algorithmes de réduction de modèle ont été développés. Il s'agit de procédures permettant de trouver des modèles réduits qui représentent la relation entrée/sortie du modèle fin de manière très précise mais en utilisant un nombre réduit d'équations.

Dans cette thèse, les techniques de réduction de modèles sont analysées et améliorées. Une attention particulière est faite à la technique de correspondance des moments. Les problèmes de placement des points d'expansion, de stabilité et de précision sont plus particulièrement étudiés. Cela a permit la simulation et conception optimale de modèles électromagnétiques complexes qu'un jeu de bus bar et un problème de diffusion d'ondes.

De plus, nous avons développé une méthode d'échantillonnage adaptatif. Des algorithmes classiques d'optimisation ont été couplés aux modèles réduits permettant l'accélération des calculs. Les méthodes proposées ont été testés dans des problèmes électromagnétiques obtenus par la méthode « Partial Equivalent Circuit Element (PEEC) ».

List of Figures

1.1 Flowchart of Numerical Modeling. Third image taken from Wang et al. (2012). 3
1.2 Example of a complex system built by the integration of simpler ones. Reproduced from (Simic et al., 2008). 5
1.3 Example of surrogate model (a) Exact function (b) Surrogate model . . . 7
1.4 Schema representing the idea behind model order reduction 9

2.1 Illustration of the cantilever beam used as an example. Adapted from Panzer et al. (2009). 28
2.2 Simulation in the frequency domain of the reduced systems obtained by different values of δ 29
2.3 Comparison of the impulse response of the High-Fidelity Model and the Reduced Order Model 29
2.4 Zoom on the comparison of the impulse response of the High-Fidelity Model and the Reduced Order Model in the time domain. 30
2.5 Sampling of the High-Fidelity System 35
2.6 Frequency response for different number of vectors in the POD basis . . 36
2.7 Depiction of the first two vectors of the subspace computation 43
2.8 Black box model for the matrix product 43
2.9 Example of the separation in (2.69) for a small matrix. 46
2.10 Frequency response for different number of moments matched 48

3.1 Scheme comparing the Classical Arnoldi process and the proposed one . 56
3.2 Scheme showing the nonzero elements of the matrix addition between an upper Hessenberg matrix and an upper triangular matrix 71
3.3 Simulation time obtained using the two different methods 72
3.4 Geometry for the loop antenna 74
3.5 Current distribution on the surface of the antenna 75
3.6 Frequency response of the Reduced Order Model. 76
3.7 Poles for the original and stabilized Reduced Order Models. 76

3.8 Time domain simulation of the stabilized system. 77
3.9 Expansion point placing for the loop antenna. (a) Placement of the points at the limits of the interval. (b)-(e) Placement of the points by the adaptive method. (f) Final position of all the expansion points. 77

4.1 Drawing of the laminated bus bar. Adapted from the original (Kuwabara et al., 2016). 80
4.2 Illustration of the power bus bar model. 81
4.3 Geometry of the capacitor 82
4.4 Flowchart of the platform integration 83
4.5 Frequency response for the laminated bus bar. 86
4.6 Time response of the system when excited by a 2 kHz square wave (a) Reduced Order Model (b) High Fidelity Model 86
4.7 Example of field vectors placement using a TM_Z Yee grid. 89
4.8 Illustration of the electric properties of breast tissue. (a) sagittal slice (b) top view 92
4.9 Time domain simulation (a) Time = 30 nS (b) Time = 48 nS 93
4.10 Frequency response obtained at the receiver for the wave scattering problem. 94

5.1 Interpolated function obtained by RBF interpolation 99
5.2 Example of the Kriging estimation 104
5.3 Example of a geodesic on a hyperbolic paraboloid 106
5.4 Frequency response of an interpolated Reduced Order Model by Subspace Interpolation. 109
5.5 Interpolated function using Subspace Interpolation. 110
5.6 Frequency response of an Interpolated Reduced Order Model by Matrix Interpolation 113
5.7 Interpolated function using Matrix Interpolation. 113

6.1 Illustration of the different sampling methods 117
6.2 Drawing of the laminated bus bar. Adapted from the original (Kuwabara et al., 2016). 120
6.3 Comparison between the original and optimized position for the capacitors. 121
6.4 Optimization results. (a) Voltage on the original system. (b) Current on the original system. (c) Voltage on the optimized system. (d) Current on the optimized system. 122
6.5 Examples of simplexes obtained by the generating method. 125

6.6 Example of reflected simplex. 126
6.7 Interpolation zone inside a simplex 127
6.8 Illustration of the system. 127
6.9 Optimization results. 128
6.10 Simplexes obtained during the evolution of the optimization point . . . 129
6.11 Comparison between the original and interpolated objective function. . 130

D.1 Circuit elements and sign convention 155

E.1 Example of two Gaussians, one with independent variables and one with dependent ones. 163

List of Tables

2.1 Time comparison between the different methods. 49

4.1 Geometric characteristics of the two plates. 82
4.2 Positioning of the discrete elements on the bus bar 82
4.3 Positioning of the discrete elements on the bus bar 85
4.4 Order and simulation time for the High Fidelity Model and Reduced Order Model. 85
4.5 Position chose for the emitter and receiver. 91
4.6 Order and simulation time for the High Fidelity Model and Reduced Order Model. 93

6.1 Number of calls by type of optimization process. 128
6.2 Results for the different optimization approaches. 129

List of Symbols

A	Principal state space matrix
E	Mass matrix of the state space
B	Input matrix of the state space
C	Output matrix of the state space
x	State vector of the High Fidelity Model
y	Output vector of the High Fidelity Model
H	Transfer function or Hessenberg matrix, context dependent
s	Laplace variable
T	Eigenvectors or coordinate transformation matrix, context dependent
z	State vector after coordinate transformation
x_r	State vector of the Reduced Order Model
y_r	Output vector of the Reduced Order Model
A_r	Principal state space matrix of the Reduced Order Model
E_r	Mass matrix of the Reduced Order Model
B_r	Input matrix of the Reduced Order Model
C_r	Output matrix of the Reduced Order Model
$\mathcal{V}$,$\mathcal{W}$	Projection Matrices
U,V,W	Matrices of two columns used to form rank-2 updates
W_0	Observability Gramian
W_c	Controllability Gramian
S	Snapshots matrix
$M_k(s_0)$	Matrix containing the k-th moment around the point s_0
K	Matrix used to generate the Krylov subspace
v	Generic vector in the Krylov subspace
r	Residue vector
D	Eigenvalues matrix
$\mathcal{L}$	Laplace transform

List of Acronyms

CEM	-	Computational ElectroMagnetics;
DAE	-	Differential Algebraic Equations;
EI	-	Expected Improvement;
FEM	-	Finate Element Method;
FIFO	-	First In First Out;
HFM	-	High-Fidelity Model;
MIPSE	-	Modeling of Interconnected Power SystEms;
MOR	-	Model Order Reduction;
OFMM	-	Orthogonalization on the Fly for Moment Matching;
PEEC	-	Partial Element Equivalent Circuit;
POD	-	Proper Orthogonal Decomposition;
RBF	-	Radial Basis Function;
RBI	-	Radial Basis Interpolation;
ROM	-	Reduced Order Model;
SISO	-	Single Input Single Output;
SVD	-	Singular Value Decomposition;

Chapter 1

Introduction

1.1 Motivation to reach very fast computer models

The understanding and modeling of the natural world is greatly related to the comprehension of some fundamental laws of physics (Feynman et al., 2013). They range from statements about the conservation of some quantity as energy or momentum to very elegant sets of equations that governs the evolution of a system in time. One example of these kind of equations is the greatly celebrated Newton's laws of motion.

These types of laws are present in all branches of physics. In electromagnetism, the famous Maxwell's equations governs the behavior of the electric and magnetic fields (Maxwell, 1881). In fluid dynamics, the Navier-Stokes equations have, under some simplifying assumptions, the same role (Currie, 2012).

All of these physical laws are written in mathematical form and very commonly in the form of partial differential equations. The framework of differential equations is then a powerful tool in understanding and modeling the world around us. However, if the system is not very simple or does not presents a useful symmetry to be exploited, it can be very difficult to solve this sets of equations even for apparently problem-free situations (Jin, 2014). Even writing down explicitly the equations may pose a big challenge for a human being.

The solution adopted in the last decades is the use of a digital computer to solve these problems. There are many methods that can be used to write down and solve these equations for very complex systems. Probably the simplest and most straight forward methods is the use of finite differences to approximate derivatives and obtain a generally sparse linear system to be solved, allowing the approximation of the solution (Strang, 2007).

Such a strategy is quite adequate for one-dimensional problems. However, it may be very hard to apply it in higher dimensions and in systems having a complex geometry or boundaries. One of the most famous approaches for dealing with these cases is the Finite Element Method (FEM) (Polycarpou, 2005). In this technique the domain of the problem is discretized and very simple and local basis functions are used to interpolate the underlaying solution. The discretization transforms the set of partial differential equations into a sparse linear system of equations (Volakis et al., 1998). This method is quite general and has obtained enormous success in structural analysis, fluid dynamics and electromagnetism.

Another alternative method that has been developed for the context of electromagnetic fields computation is the Partial Element Equivalent Circuit (PPEC) (Nitsch et al., 2009). This technique also discretizes the space and obtains an equivalent circuit whose variables can be directly correlated to values of the physical quantities of the modeled system. Differently from the FEM method, the matrices involved in the modeling are not sparse. However, there are matrix compression techniques that can be used to reduce the required storage (Paixão and Coelho, 2015).

The Partial Element Equivalent Circuit method converts partial differential equations into ordinary differential equations by discretization of the space. Although these kind of equations are simpler to solve, they can still be quite a challenge if they come in big numbers. If the geometry of the system is very complex or if the accuracy required for a given application is too high, there could be hundreds of thousands of simultaneous equations to be solved. Having discussed these issues, one can look at Figure 1.1 and identify the steps of numerical modeling.

It is tempting to assume that these models can be broadly used to any kind of applications. However, this is not true. If the envisaged application is, for example, the optimization of an electric motor, each evaluation of the objective function will demand a complete simulation of a slight different motor for each needed parameter set. This can be very time consuming, limiting the practice of this kind of task to model only of moderated size or to problems that time is not a immediate constraint, like the planning of electrical networks, which can be done with more than 20 years in advance (Seifi and Sepasian, 2011).

It is exactly in this point that Model Order Reduction (MOR), the main theme of this thesis, comes in. Instead of solving this huge number of equations in a very long computational process, one can look for a smaller number of equations that mimics quite well the dynamical behavior of the original system (Antoulas, 2005).

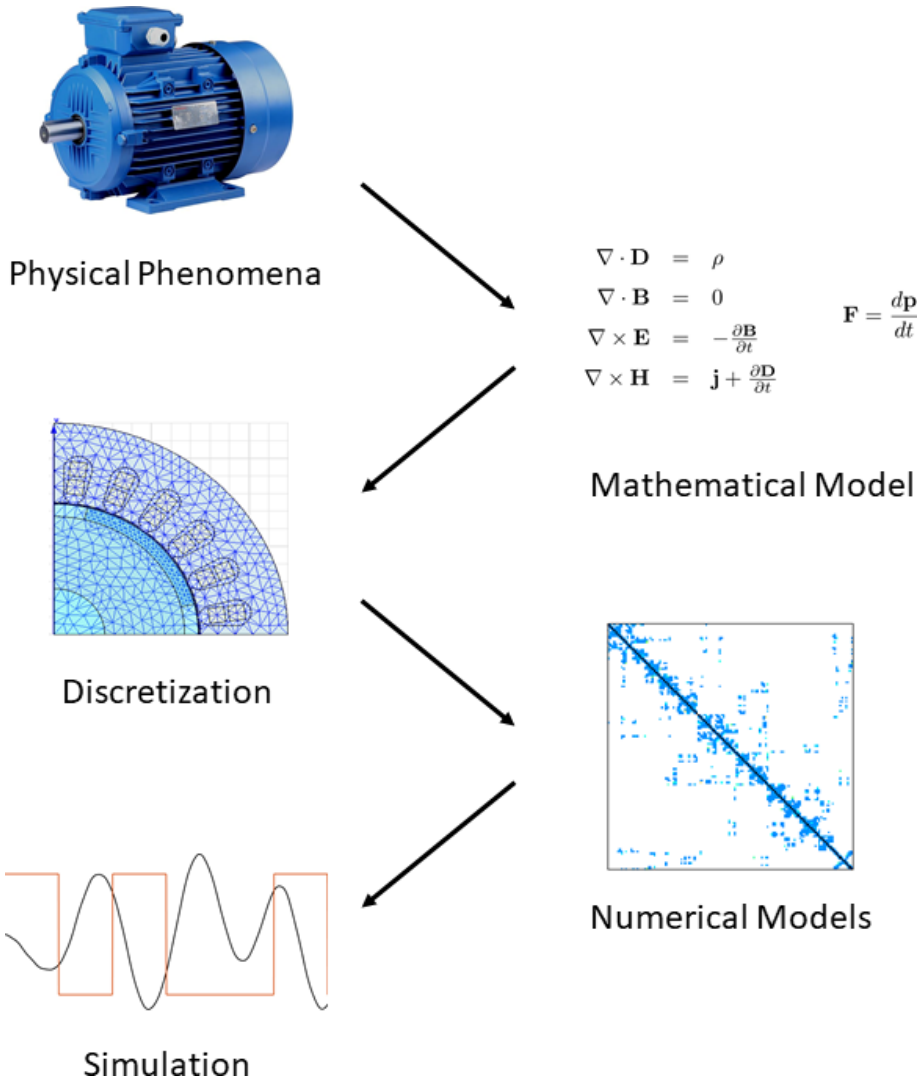


Figure 1.1: Flowchart of Numerical Modeling. Third image taken from Wang et al. (2012).

1.1.1 Simulation of Electromagnetic Devices

It should not be hard to convince someone that the developments in electromagnetic theory has had a drastic impact in our daily lives. Examples range from electrical household appliances, like washing machines and refrigerators, to less familiar ones as antennas, integrated circuits and electrical motors.

Basic understanding of electromagnetic equipment can be obtained directly from the application of the Maxwell's equations followed by a set of simplifying hypothesis. However, the development of digital computers have allowed the analysis of more

complex devices and originated a new field called Computational Electromagnetics (CEM) (Bondeson et al., 2005) (Coulomb and Sabonnadière, 1985). This allows the accurate simulation of complex electromagnetic phenomenon like flux leakage and skin effect in high frequencies.

One of the principal tools of CEM is the Finite Element Method, initially proposed to solve equilibrium and vibration problems (Courant et al., 1943), it has quickly been adapted to the electromagnetic framework and was successfully used to many different applications. In the late sixties it has been used to perform waveguide and cavity analysis (Silvester, 1969). In the sevenths it started to be used in problems in magnetostatic (Sarma, 1976), the analysis of electrical machinery (Demerdash and Nehl, 1979) and even in geophysics (Coggon, 1971) applications.

The following years were filled by advances in the meshing techniques and in applications of this method with optimization routines (Garcia and Webb, 1990). The method has greatly attracted the attention of the numerical simulation community in the 80's and 90's and has since become a standard (Polycarpou, 2005).

More recently, Partial Element Equivalent Circuits (PEEC) techniques have been popularized due to its successful application in compact devices and its easy physical interpretation (Ruehli, 1974) (Roudet et al., 2007). This method is also particularly interesting to the analysis of transmission lines (Nitsch et al., 2009). It consists in the discretization of the domain, like the finite element method, and use of an equivalent circuit in each of these divisions. This will result in a circuit that can be solved by any classical circuit solver.

In this thesis, we will focus mainly in applying the Model Order Reduction framework to allow fast simulation of the kind of systems mentioned above. One very important instance were simulation of electromagnetic devices is crucial is integrated system simulation. It consists in simulating one complex system composed by smaller subsystems that may have been obtained by different methods and may have different physical formulations. Therefore, it is important to obtain fast and accurate models to reach acceptable computational times. Figure 1.2 shown an example of such a system (Simic et al., 2008). In this example the motor MG2 can be obtained by the use of a electromagnetic model.

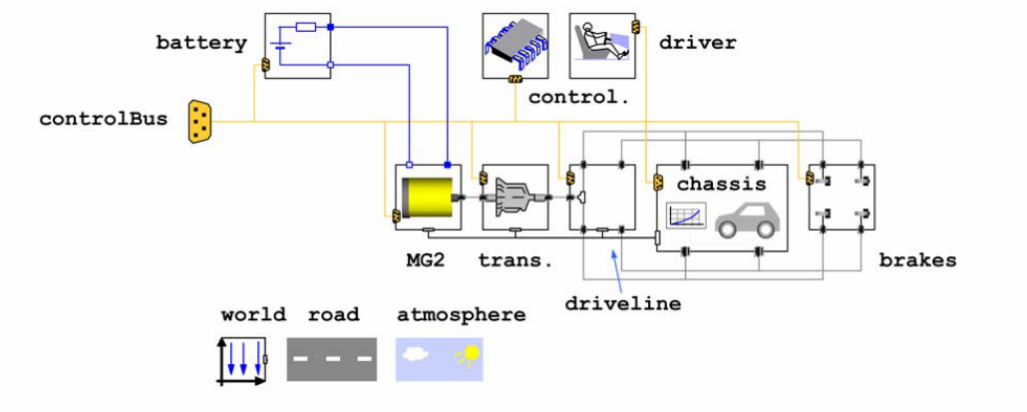


Figure 1.2: Example of a complex system built by the integration of simpler ones. Reproduced from (Simic et al., 2008).

1.1.2 Optimization of Electromagnetic Devices

One of the main uses of the electromagnetic models introduced in the last sections is the optimization of a given device. For this kind of application, one or many objective functions are described by some performance criteria that the system must attain. In order to do this, one must simulate the High Fidelity Model (HFM) many times, at least once for each point analyzed by the optimization algorithm. As said before, this can be very time consuming and sometimes not practical due to time constraints.

To overcome this difficulty many strategies were developed in order to allow optimization with models having high dimensionality or whose simulations takes a long time. Some of these strategies are presented in the following sections.

Surrogate Models

One of the natural approaches to reduce simulation time is to use a surrogate model. This means that a coarse model is used instead of the original one, allowing quicker simulations at the cost of losing accuracy. Such a model can be obtained, for example, if a coarser grid is utilized in the discretization step (Bandler et al., 1994a), or by some physical knowledge of the physical problem. In (Bandler et al., 1994b), for example, a simple circuit model is used instead of the field-theoretic model and the circuit parameters are adapted to obtain responses similar to the original system.

Another approach is to use model identification to obtain a transfer function of low order that approximates the behavior of the full model direct from simulation data.

There are many ways of doing this and many different approaches can be found in the literature. In (Prando and Chiuso, 2015) a bayesian inference approach is used to identify a system while keeping the dimension of the result as low as possible. In (Peherstorfer and Willcox, 2016), the author obtains a reduced order model by direct identification of the transfer function of the high fidelity model by a series of samples obtained for different set of parameters of the original system.

An approach to obtain approximate solutions for optimization is called Space Mapping. This method does not have as its objective the construction of only one accurate model to represent the system. It performs adaptive steps in order to change the low-dimensional model parameters until convergence is reached at an optimal point (Bakr et al., 2000). On an intuitive level, one can image a Quasi-Newton strategy that performs iterative updates to the Hessian matrix until convergence is reached at the, hopefully, optimal solution.

This strategy has been successfully utilized in electromagnetic design optimization (Ayed et al., 2012). In this work, the authors propose a modification on the classic approach, using three granularities instead of two. The result is the reduction of computational time and reliable results.

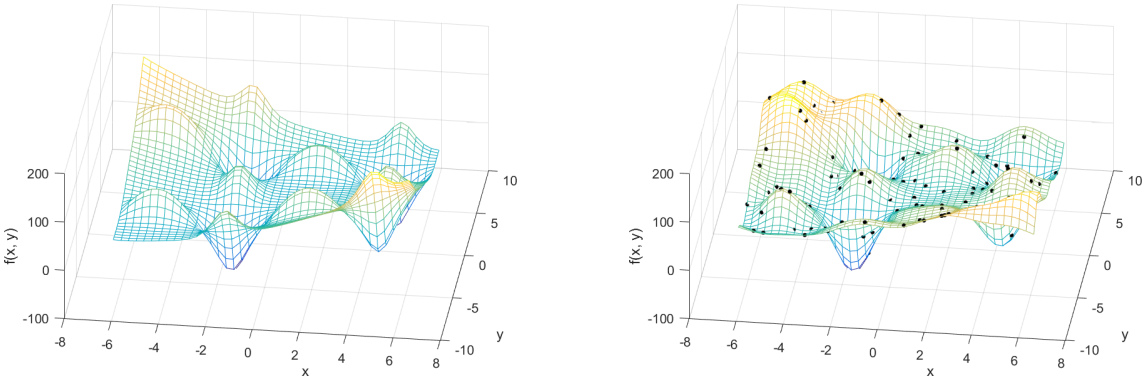
In Delinchant et al. (2013), the authors have compared different space mapping approaches in the optimization of magnetic devices. Their work leads to the conclusion that the manifold mapping technique is a very promising technique to accelerate the computational time of optimizing complex magnetic systems.

Interpolation Methods

The methods presented until this point aim at producing a system that is capable of reproducing the dynamic of the original one but with a lower computational cost. If the objective of this computation is optimization, however, one can focus the attention only to the result of the objective function and try to interpolate this value directly from a small number of actually computed set of parameters. In the following paragraphs different interpolation techniques, that can be used for the optimization process, are described.

However, before proceeding, we present an intuitive example of what can be accomplished with this approach. Figure 1.3 presents the comparison between the exact optimization function and its surrogate model obtained by Kriging, one of the methods

that will be discussed shortly. This function is known in the optimization literature as the Bird Function (Jamil and Yang, 2013). The black points represent sampling points whose values are used to extrapolate the remaining ones.



(a) (b)

Figure 1.3: Example of surrogate model (a) Exact function (b) Surrogate model

It is known that for a set of distinct $n+1$ points, there is one and only one polynomial function of order n that interpolate all these points. It is then tempting to assume that using a polynomial with a high enough order will be able to produce good interpolation of any data set. This is however, very far from the truth. Polynomials of high degree tend to have very erratic behavior between the sampling points and therefore have very low predictive power (Press, 2007).

However, one can fix the order of the polynomials and construct local interpolation functions for each region of the space and impose some smoothness conditions that they should satisfy. This is the main idea of Splines and most commonly cubic-splines (Press, 2007). A downside of this method, is the requirement of a grid of points all over the space.

Interpolating points without a grid structure is known as the problem of scattered data interpolation (Wendland, 2004). One important aspect of the methods that deal with this problem is that they are easily extended to any number of dimensions.

Radial Basis Interpolation

Radial Basis Interpolation (RBI) consists of a very clever and simple mathematical idea that can be used to perform function interpolation of scattered data in arbitrary dimension. It assumes that each sampled point has an influence on its neighbor points

represented by a function of the distance from the sampled point (Press, 2007). This leads to a linear system whose solution allows the approximation of unsampled points in the search space.

Radial Basis Functions are quite popular in the system identification community (Warwick and Craddock, 1996). They also have many applications in Neural Networks (Franco and Steiner, 2017). They can also be integrated into optimization processes to produce effective global search methods (Qiang et al., 2011).

Kriging

Another very powerful method of scattered data interpolation is Kriging (Isaaks et al., 1989). It assumes that each evaluated point is the result of a sampling from a probability distributions and finds out the other values by minimizing the variance of the distribution and requiring zero mean error. One of the main advantages of this technique is that the variance of the error at an unknown point can be computed. Of course, the accuracy of the obtained value will be function of the quality of the model chosen to estimate the correlation among the sampled points. Chapter 5 will present one such model based on (Sacks et al., 1989).

This interpolation technique has been used in optimization processes, as in (Xia et al., 2014) where it has been used in the multi-objective optimization of electromagnetic devices. Another example is (Rudolph and Stitt, 2015) where the authors use the Kriging method to the optimization of complex systems.

One particularly successful algorithm involving Kriging is ParEGO (Knowles, 2006). It uses the combined values of the objective function and of the estimated variance of the error to select promising points in the parameter space. This metric is known as the Expected Improvement (EI).

This algorithm also makes use of a procedure named DACE (Design and Analysis of Computer Experiments) [1] (Lophaven et al., 2002). In chapter 5 we present this topic in detail.

Having made an overview over surrogate methods, we present different techniques to perform Model Order Reduction in the next section. Note that the Reduced Order Models obtained may also be used for optimization purposes. Besides that, they can be used to perform time domain and frequency simulations of the systems.

[1] http://www2.imm.dtu.dk/projects/dace/

1.2 Different Model Order Reduction Approaches

Model Order Reduction (MOR) is a set of techniques that seek a small set of differential equations that are capable of reproducing very well the behavior of the original High-Fidelity Model. The resulting small number of equations can be solved by any classical solver in an acceptable computational time. Figure 1.4 shows an schematic representation of this idea. Note that both inputs and outputs are unchanged.

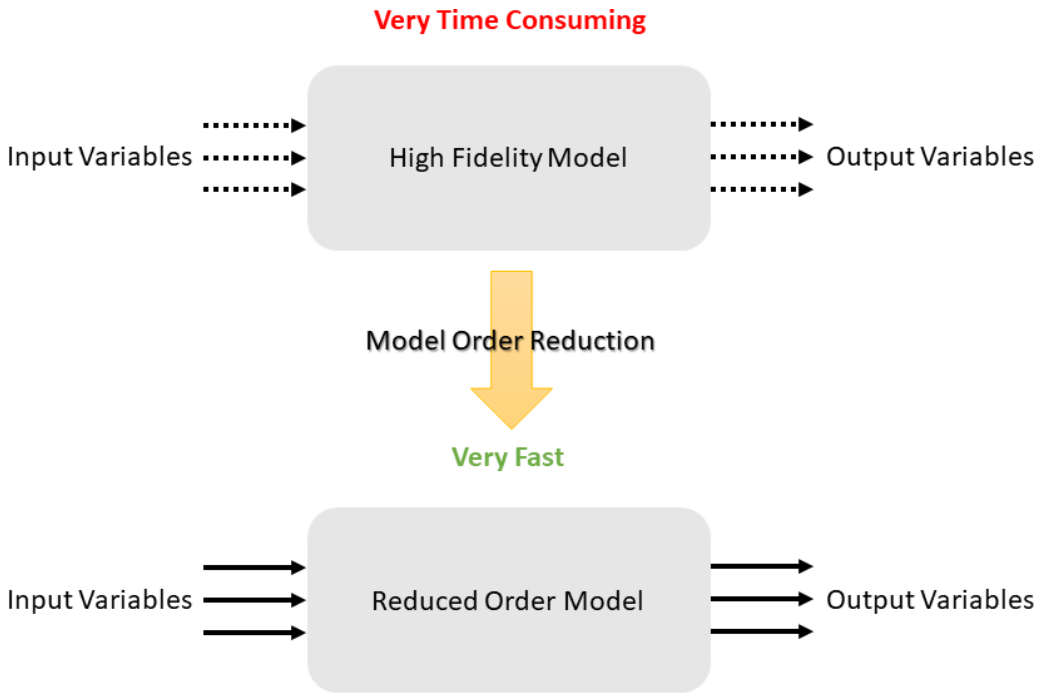


Figure 1.4: Schema representing the idea behind model order reduction

In this section, we present the classical and well established techniques in order to perform Model Order Reduction. The mathematical foundations of some of them are presented in details in Chapter 2.

1.2.1 Vector Fitting

Vector fitting [2] is a class of techniques that interpolates a transfer function directly from measures in frequency domain of the original system (Gustavsen and Semlyen, 1999). This is done by the identification of some poles and residues by the solution of a least squares problem.

[2] https://www.sintef.no/projectweb/vectfit/algorithm/

Recent developments have allowed better convergence and results that are more robust to the initial choice of the poles (Gustavsen, 2006). It is also possible to write the result in the form of a state space system (Semlyen and Gustavsen, 2000). Finally, even passive models can be obtained by the solution of a quadratic programming problem (Gustavsen and Semlyen, 2001).

1.2.2 Balanced Truncation

One of the best known Model Order Reduction techniques is balanced truncation (Moore, 1981). It consists in the elimination of states that do not contribute in a significant manner to the input/output behavior of the system.

This method is known for its accuracy and to preserve stability in the ROM if the original HFM has this property. Other than that, this method provides a rigorous error bound for the frequency response of the ROM (Enns, 1984).

There is however, an important drawback in the application of this method. It requires the solution of Lyapunov equations that require $O(n^3)$ work (Bartels and Stewart, 1972), n being the size of the problem. Another limited application of the method is that it can only be applied to linear models.

1.2.3 Proper Orthogonal Decomposition

The Proper Orthogonal Decomposition (POD) has received many names in different fields of applications. In the statistics community, for example, it is known as principal component analysis (Jolliffe, 2002). In the framework of Model Order Reduction, it consists basically as an empirical method to obtain projection subspaces that represent very well the manifold that the solution of the original problem lives in (Chatterjee, 2000). Some advances have been done in the direction of finding a relation between proper orthogonal decomposition and an approximative form of balanced truncation (Willcox and Peraire, 2002).

This method has the advantage that it can be applied to nonlinear models with the cost of the simulations of some of the dynamics of the original system (Henneron and Clenet, 2014). It cannot, however, guarantee that the reduced models are passive or stable.

1.2.4 Moment Matching

Another kind of Model Order Reduction approach is Moment Matching (Freund, 2000). It is also known as Krylov subspace method due to the fact that main step of the computation of the projection matrix is actually the computation of a very special Krylov Subspace. This method is known to perform well when the dimension of the original system is very high (Antoulas, 2005).

One of the main advantages of this method is that it only requires the product of the system matrices to an arbitrary vector. Therefore, if a matrix compression technique is used it is not an obstacle to its application. This would be, for example, the case when dealing with integral formulation of the Maxwell's equations like PEEC that produce dense matrices that are normally too big to be stored in memory directly (Siau, 2016). In Chapter 6 we will encounter one example of such a system.

The method has, however, some problems in its original form, as the possible loss of stability and the complex system matrices that result for complex expansion points. This is not a problem in frequency analysis, but is crucial for the time simulation and integration of these kind of methods to external system simulation softwares, which are problems that we are willing to solve.

1.3 Objectives

In this section, the main objectives of the thesis are listed and individually discussed. They can be separated in two big categories : model order reduction and optimization. Actually, the final objective of the thesis is to show the efficiency of MOR in order to perform fast dynamic simulation in both frequency and time domain, for system simulation and optimization of complex electromagnetic device. Note that having accurate systems for time domain simulation is also directly connected to the integrated simulation of complex systems.

During the development of the work, a deep comprehension of the reduction techniques has became of main interest and has been extensively developed.

There are some well established algorithms to perform MOR. Some of them, like balanced truncation, have been known to work extremely well. What is then the

interest in doing further research in this field ? The answer can be found by analyzing the computational time and limitation of each of the algorithms.

Techniques that produce very accurate and stable models are normally computationally very expensive. There are other methods like Moment Matching and Proper Orthogonal Decomposition that have a smaller computational burden but do not present some physical properties of interest. Another problem is the presence of degrees of freedom that are problem specific posing difficulties if one tries to build a general algorithm.

In this thesis we have solved some of the problems faced by the use of the Moment Matching technique to reduce High Fidelity Models. The main concern when reducing a model is that its dynamical behavior should approximate very well the one of the original model.

It is not an easy task to assure that the reduced process will produce accurate results for an arbitrary input model. If someone has a kind of application in mind, it could use the physical knowledge of the model to set the reduction parameters and obtain moderately accurate solutions.

In this thesis, however, we deal with Model Order Reduction in a broader manner. We do not assume a specific field of application, even though our main focus is electromagnetic simulation. The developed algorithm should work as all for electrical machines as for structural dynamics.

In order to solve this big challenge, adaptive approaches have been developed. They assure that the model has a good accuracy for a given frequency range without assuming any particular kind of shape for the transfer function.

The process of projection of the High Fidelity Model into a space of smaller dimension do not guarantee that the physical properties of the original model will be preserved. A stable model can became unstable when reduced. One can argue that for frequency domain simulations this is not an issue if the system response is adequate inside a given frequency range. However, if time simulations are envisaged, unstable models cannot be dealt with. During the solution of the differential equations, divergent behavior will dominate the solutions.

Finding solutions to this issue has been a problem in the model order reduction community for the last decades. Some methods have been developed for specific physical phenomena like fluid flow or electrical circuits (Freund, 2011a). More general

methods have been proposed recently like (Bond and Daniel, 2008a) and (Amsallem and Farhat, 2012). However, these methods demand the solution of optimization problems that are numerically ill conditioned or are not very efficient in the computational point of view.

One of the main contributions of this thesis is the introduction of a new method to stabilize any projection based reduced order model. The proposed technique shifts from the control engineering framework of Lyapunov Equations and Positive Definite Lemas and works with pure linear algebra concepts as eigenvalues and rank-1 updates. This approach allows the construction of a method that is the most general one known by the author. All the needed computations are performed into the reduced matrices, therefore, the method is computationally inexpensive.

In the field of system modeling and MOR some of the most fundamental facts about digital computers becomes very evident : they work in finite precision arithmetic and take a finite amount of time to execute each computation. This means that some of the mathematical facts and algorithms are simply not suitable to direct implementation. An example is the solution of Lyapunov Equations that in theory can be casted into solving a linear system. However, the computational complexity is $O(n^6)$, where n is the size of the problem. Therefore, the straight forward mathematical method would take longer than one is whiling to wait to computing the solution. Another example, that will be studied exhaustively in this text, is the computation of the Krylov subspace.

In this thesis numerical robust algorithms are presented to compute the reduced order models. Some of the main contributions is in the way that we avoid complex arithmetic and how we manage to compute subspaces that are orthogonal even if many expansion points are chosen. These two points are very important if the system is to be coupled to external simulation tools and if good accuracy is needed for a large frequency range.

1.4 Structure of the Thesis

This thesis is composed of 7 chapters and appendices. The current chapter has introduced the subject of research and the challenges that one may face when dealing with Model Order Reduction. The second one consists in a presentation of well established MOR techniques. The third chapter contain contribution that we have made in

the field of MOR by Moment Matching. Examples of reduction of complex systems are given in chapter. A more in-deep view of optimization with very costly models is made in chapter 5. Chapter 6 presents some results in optimization of a real world problem using the developed techniques. Finally, a conclusion is presented in chapter 7, ending the main text.

The heavier mathematical profs and concepts can be found in the appendices. This way, we try to keep the reading as fluid as possible and in the same time present the mathematical tools needed for really understanding how the algorithms work.

1.5 A Word on Notation

This brief section will clarify some basic conventions that are adopted throughout this thesis in order to avoid any easily solved misunderstanding in the equations.

The real will be denoted by $\mathbb{R}$. If superscripts are used, they represent the Cartesian product of this set by itself. In other words, $\mathbb{R}^s$ means the same as $\mathbb{R} \times \mathbb{R} \times \ldots \mathbb{R}$ a total of s times. The symbol $\mathbb{C}$ represents the set of all complex numbers. The real part of a number of denoted by $\mathfrak{R}$ and its imaginary part is given by $\mathfrak{I}$. A * is used to exclude the zero from the set. Similarly, a $_+$ excludes the negative numbers. Therefore, $\mathbb{R}_+^*$ represents all positive real numbers bigger than zero.

Transposition of a matrix A is denoted by A^T. This means that $(A)_{ij}^T = A_{ji}$. The conjugate of the transpose of A is represented by A^*. In index notation this can be understood as : $(A^*)_{ij} = A_{ji}^\dagger$. This last equality gives the opportunity to introduce a new symbol $A^\dagger$ that represents the conjugate of the matrix A. If a matrix is complex the conjugate of the transpose appears more naturally in the equations. In some situations, however, this is not the case and we really mean simple the transposition of a complex matrix.

All vectors are represented by lower case letters and are assumed to be column vectors. Therefore, the transpose of a vector will aways be a line vector. This allows the immediate recognition of expressions like $x^T Ax$ or $u^T v$ as valid scalar quantities, whereas expressions like uv or xA are not valid. If x is a vector, then we denote a generic entry of this vector as x_i.

Sometimes we use a dot to represent time derivative and an apostrophe to represent derivative in respect to the only dependent variable. This is in conformation with most calculus text books and should appear as being natural to the reader.

In a more unorthodox fashion, ∇_x is a symbol that means the derivative of an expression with respect to each one of the entries of a vector x. Therefore, the application of this symbol to a scalar produces a new column vector. Using a similar reasoning, its application to a line vector results in a matrix.

1.6 Conclusion

In this chapter we have introduced the problem of simulating and optimization of complex electromagnetic devices. A brief review of the literature containing methods capable of dealing with such situations has been presented. No mathematical details of any method has been given.

The following chapters will develop in further detail the methods mentioned in the introduction. Many examples of applications will be presented.

Chapter 2

Constructing Reduced Order Models

2.1 Introduction

This chapter contains the mathematical details of some well established Model Order Reduction techniques. This is necessary to understand the context in which the contributions of this thesis are inserted.

These methods are Balanced Truncation, Proper Orthogonal Decomposition and Moment Matching. Each one of them will have specific advantages and drawbacks. Special attention will be paid to Moment Matching, since it will be the main target of the next chapter.

2.1.1 The State Space Representation

Before reducing a model, one has to define which kind of model is dealt with. There are many different ways to model a physical system. One of the most successful approaches is the modeling by differential equations. Many physical laws are naturally written in these terms. In this thesis, these differential (algebraic) equations will be assumed to have the form (2.1). In this equation: $A, E \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times e}$, $C \in \mathbb{R}^{s\times n}$. The vector $x \in \mathbb{R}^n$ is called the state vector and $y \in \mathbb{R}^s$ is the output vector. Finally, $u \in \mathbb{R}^e$ is the input vector. This form is called the state space representation of a dynamical system (Chen, 1995).

$$\begin{cases} E\dot{x} &= Ax + Bu \\ y &= Cx \end{cases} \tag{2.1}$$

This representation of a linear dynamical system is very useful and allows a variety of physical interpretations of each of its matrices (Friedland, 2012). If the matrix E is singular, this set of equations is said to be in descriptor form and represents a set of Differential-Algebraic Equations (DAE). In an intuitive way, this means that the derivative for each entry of the state vector cannot be written explicitly.

It is very common to find control engineering text books with an alternative form for (2.1), given by (2.2) (Chen, 1995) (Hespanha, 2009). In this alternative presentation, the matrix E is assumed to be the identity matrix. This means that the derivative for each of the states is known in an explicit way.

$$\begin{cases} \dot{x} &= Ax + Bu \\ y &= Cx \end{cases} \tag{2.2}$$

This form can be obtained by solving the linear systems $E^{-1}A$ and $E^{-1}B$. Nevertheless, this operation can be very costly or even impossible if E is singular. Which form is the most natural one will depend entirely on the problem. However, it is normally a good idea to preserve the E matrix and do not perform the inversion step.

As normally done to a set of differential equations, another very useful way to represent them is obtained by the application of the Laplace transform, denoted by $\mathcal{L}$. Doing so, the system in (2.1) can be written as in (2.3). The scalar s is the Laplace variable is normally chosen to be purely imaginary to allow a physical interpretation of the result. It is also important to say that in this kind of analysis, one considers the initial conditions of the systems as being all zero.

$$\begin{cases} sE\mathcal{L}\{x\} &= A\mathcal{L}\{x\} + B\mathcal{L}\{u\} \\ \mathcal{L}\{y\} &= C\mathcal{L}\{x\} \end{cases} \tag{2.3}$$

Using this way of writing the equations, one can identify a matrix $H(s)$, called the transfer matrix of the system (Luenberger, 1979), that relates the Laplace transform of the inputs into the Laplace transform of the outputs. Its expression is given by (2.4).

If this system has only one input and one output, this matrix will degenerates into a scalar that is known as the transfer function of the system. For the more general case, the entry (i, j) of this matrix represents the transfer function for the i-th input to the j-th output.

$$H(s) = C(sE - A)^{-1}B \tag{2.4}$$

This way of looking at a dynamical system given in descriptor form will prove to be very useful in the subsequent development. Arguably, the biggest advantage of this form is that its analysis requires solving a linear systems while the original problem demands the solution of a set of differential equations.

It is natural to wonder if H is uniquely determined by a unique set of matrices A, B, C and E. Can different matrices produce the same transfer function ? The answer is yes and will have an important application in Model Order Reduction. A particular set of equations is called a realization of the system (Fairman, 1998). This nomenclature is justified because many different matrices can produce the same dynamical behavior and, consequently, the same transfer function. To ilustrate this fact, one can chose a non-singular matrix T and define a change of variables as in (2.5).

$$x = Tz \tag{2.5}$$

This new state space vector z, can be directly used to express (2.1) in an equivalent way given by (2.6). Notice that the multiplication by the inverse of the transformation matrix is not mandatory but is a very natural way to proceed. If the matrices A and E are symmetric or positive definite, for example, not performing this multiplication would result in lost of these properties.

$$\begin{cases} T^{-1}ET\dot{z} &= T^{-1}ATz + T^{-1}Bu \\ y &= CTz \end{cases} \tag{2.6}$$

Using these new matrices to compute the new transfer function will result in a set of simplifications that will recover the original transfer function of the system before the change of variables, as one can see in (2.7)

$$H_T(s) = CT(sT^{-1}ET - T^{-1}AT)^{-1}T^{-1}B = CTT^{-1}(sE - A)^{-1}TT^{-1}B = H(s) \tag{2.7}$$

This short introduction to the state space representation is far from being complete but should give the reader all the background to understand the following sections.

Discrete State Space Representation

So far, we have described the state space when only space has been discretized. Therefore, the dependency in the time derivative is still present.

If one continues the process of discretization and applies it to the time, the state space will be converted from a differential equation into a recurrence equations given by (2.8).

$$\begin{cases} E_d x_{k+1} &= A_d x_k + B_d u_k \\ y_k &= C_d x_k \end{cases} \tag{2.8}$$

The relations between the matrices in the continuous and discrete systems are dependent of the employed discretization method. For a quick but very enlightening discussion of this subject, the reader can refer to (Chen, 1995).

As we did for the continuous system by employing the Laplace transform, the z transform (Oppenheim et al., 1997) can be applied to this set of equations and will produce a transfer function given by (2.9).

$$H(z) = C_d(zE_d - A_d)^{-1}B_d \tag{2.9}$$

Note that both results are extremely similar.

2.1.2 Model Order Reduction by Projection

The simulation of a system may be understood as the solution of (2.1) for a number of different inputs or the computation of the matrix H in (2.4) for a given frequency range. There are many commercial and even open source softwares that can perform this task. However, as the number of variables grows, the computational burden can be very high and simulation time may be prohibitive.

The main idea of Model Order Reduction is to manipulate these equations describing a High-Fidelity Model and obtain a much smaller set of equations that is very efficient in representing the input/output behavior of the original model (Antoulas, 2005). Once that the reduced model has been obtained, it can be used in the place of the original one for many different applications like simulation and controller design. It is then clear that the time used in the reduction process should be very small compared to the time of using the original system directly in the application.

There are many well established ways of doing this like Asymptotic Waveform Evaluation (AWE) (Chiprout and Nakhla, 1994) and Padé via Lanczos (Cullum et al., 2000). However, in this thesis we will focus on the framework of projection based Model Order Reduction. The main idea of this class of methods is to approximate the high-dimensional state space vector x by a linear combination of a few vectors that are supposed to represent accurately the manifold that the solutions is contained in. This can be written in a more formal way as in (2.10).

$$x \approx \mathcal{V} x_r \tag{2.10}$$

In this equations, $\mathcal{V}$ is the matrix whose columns will be used as a basis to approximate the original state space vector and x_r is the reduced state space vector that contain the coefficients of this linear combination.

Using this approximation, the differential equation in (2.1) can be written as in (2.11). One must note that the result of using an approximation instead of the exact solution is that a residue term, denoted by ϵ, will be present.

$$E\mathcal{V}\dot{x}_r - A\mathcal{V}x_r - Bu = \epsilon \tag{2.11}$$

In order to eliminate ϵ from this equation, one can look for solutions that satisfies the condition $\mathcal{W}^T\epsilon = 0$, where $\mathcal{W}$ is a matrix of appropriated size. This is equivalent to projecting (2.1) into the span of $\mathcal{V}$ and orthogonally to the the span of $\mathcal{W}$, producing the set of equations in (2.12) (Panzer et al., 2010). The new system is the projected system or the reduced system. It is important to mention that the vector y_r has the same dimension as the original output vector. In this context, we use the index r to indicate that this is not the exact output but an approximated one.

$$\begin{cases} \mathcal{W}^T E\mathcal{V}\dot{x}_r &= \mathcal{W}^T A\mathcal{V}x_r + \mathcal{W}^T Bu \\ y_r &= C\mathcal{V}x_r \end{cases} \tag{2.12}$$

In order to keep the notation as simple as possible, one can define a set of matrices related to those of the original HFM as shown in (2.13).

$$\begin{array}{rclrcl} A_r &=& \mathcal{W}^T A\mathcal{V} & E_r &=& \mathcal{W}^T E\mathcal{V} \\ B_r &=& \mathcal{W}^T B & C_r &=& C\mathcal{V} \end{array} \tag{2.13}$$

Using this notation, one can simplify (2.12) into a more esthetically pleasing equation (2.14). This final set of equations represents the Reduced Order Model (ROM).

$$\begin{cases} E_r\dot{x}_r &= A_r x_r + B_r u \\ y_r &= C_r x_r \end{cases} \tag{2.14}$$

Notice that in this equation the input vector is the same as the original matrix, this means that no adjustments have to be done into the input signal in order to use the

reduced model. The same is true for the dimension of the output vector. The most important feature, however, is the size of the matrices A_r and E_r that is $q \times q$, considering q the number of columns of the matrix $\mathcal{V}$. For any reasonable reduction method $q \ll n$. Hence, the solution of the reduced system should be much faster than the original one.

The matrix $\mathcal{W}$ is often chosen to be equals to $\mathcal{V}$. This means that an orthogonal projection has been chosen. Although this is the first and most natural way of projection, for the application of model order reduction, choosing different $\mathcal{W}$ matrices can be useful to impose desired physical properties like stability (Bond and Daniel, 2008a).

The question that must be answered now is how to choose the matrices $\mathcal{V}$ and $\mathcal{W}$. It must be clear for the reader that is highly unlikely to obtain good results if random matrices are chosen. One alternative may be the use of physical insight of the problem to build some modes that represent very well the state vector of the system. This approach is quite elegant but is not general. Next section will present the methods that have been developed to tackle this problem for an arbitrary set of linear differential equations.

2.2 Techniques to Building Reduced Order Models

In this section, the somehow classic methods for Model Order Reduction are discussed. Each one of them have strong and weak points. The first one, balanced truncation, uses deep knowledge of state space systems in order to obtain really good approximations. However, its computational cost may be very high making it very difficult to use in practice with models that have a huge number of equations.

The other methods, Proper Orthogonal Decomposition and Moment Matching can be used to larger systems but have some downsides like lack of stability for the reduced order models.

In the following subsections all these three methods will be explained in a way that the reader should be able to understand them in a self contained manner. All the algorithms have been implemented in Matlab and will be compared when applied to a simple cantilever beam model, obtained by a finite element formulation.

2.2.1 Balanced Truncation

The original method of balanced truncation (Moore, 1981) was developed to deal with state space systems in the form of (2.15). Its generalized form (Stykel, 2002) only came many years later and uses a more sophisticated version of the arguments that were originally presented. To keep the exposition as simple as possible, only the original technique is presented.

$$\begin{cases} \dot{x} &= Ax + Bu \\ y &= Cx \end{cases} \tag{2.15}$$

The idea behind balanced truncation is to ignore system states that are, in a specific sense, not important to its dynamics. One needs, then, to define a quantitative measure of importance. One way that this can be done by looking at the reachability (a concept closely related to controllability (Hespanha, 2009)) and observability of each system state.

We start by looking at the solution of the differential equations in (2.15). It can be obtained by the use of the matrix exponential (Luenberger, 1979) as shown in (2.16). In this equations, $X \in \mathbb{C}^{n \times n}$.

$$e^X = \mathcal{I} + X + \frac{X^2}{2!} + \frac{X^3}{3!} + \ldots + \frac{X^k}{k!} + \ldots \tag{2.16}$$

Notice that it has the expected property that its derivative is equal to itself. Using this matrix and some algebraic manipulation, the solution of (2.15) can be found to be (2.17) (Chen, 1995).

$$x(t) = e^{At}x_0 + \int_0^t e^{A(t-\tau)}Bu(\tau)d\tau \tag{2.17}$$

This result will be used to measure the energy Ψ that can be recovered by the output of the system with no inputs and a certain initial state x_0. This value can be used as a measure of how observable the state x_0 is. In other words, how much this state can be detected by only looking at the output of the system. Equation (2.18) shows how to compute this value.

$$\Psi = \int_0^t y(\tau)^T y(\tau) d\tau = x_0^T \left[\int_0^t e^{A^T \tau} C^T C e^{A\tau} d\tau \right] x_0 \tag{2.18}$$

The matrix W_0 is called the observability Gramian and can be identified from the inner product present in the right hand side of (2.18). Its expression is written in (2.19).

$$W_0 = \int_0^t e^{A^T \tau} C^T C e^{A\tau} d\tau \tag{2.19}$$

A similar argument will be used to find a numerical value for the reachability of a given state. In order to do this, one can measure the energy needed to drive the system from its zero state to a given state $x(t)$ at a given time t. The input signal that produces such a result is given by (2.20). Although its expression may seems complicate, one can recognize this expression as being the right inverse of the integral in (2.17) followed by the desired state. One other way to convince yourself that this is actually the desired input is to plug it into (2.17) and obtain the desired state space vector at time t.

$$u(\tau) = B^T e^{A^T (t-\tau)} \left[\int_0^t e^{A(t-\tau)} B B^T e^{A^T (t-\tau)} d\tau \right]^{-1} x(t) \tag{2.20}$$

The energy Γ of this input signal can be computed in the same way as done for the observability and is given by (2.21).

$$\Gamma = x^T(t)\left[\int_0^t e^{A(t-\tau)}BB^T e^{A^T(t-\tau)}d\tau\right]^{-1} x(t) \tag{2.21}$$

The reachability gramian W_c can then be identified as in (2.22). To obtain this form, a simple change of variables in the integral was performed. It is not out of place to define the reachability Gramian as being the inverse of the quantity in (2.21). The less energy a state needs to be produced the more reachable it is considered.

$$W_c = \int_0^t e^{A\eta}BB^T e^{A^T\eta}d\eta \tag{2.22}$$

Having identified these two Gramians, the balanced truncation method finds a change of variables of the state space that yield both Gramians equal and diagonal. With this properties, one can easily eliminate from the systems states that are at the same time difficult to observe and difficult to reach.

The first step into this path is understanding how the Gramians will change face to a change of variables given by a matrix T. This result can be obtained by applying this change of variables in the definition of the Gramians. It will be then necessary to use the following property of the matrix exponential given by (2.23). This property can be directly deduced from the definition of the matrix exponential given by (2.16).

$$e^{T^{-1}XT} = T^{-1}e^X T \tag{2.23}$$

With the aid of this property it is also easy to deduce that the new gramians are given by (2.24).

$$\widetilde{W_o} = T^T W_0 T \quad \widetilde{W_r} = T^{-1}W_r T^{-T} \tag{2.24}$$

Following the method in (Laub et al., 1987), we define the singular value decomposition of the product of the lower Cholesky terms of the Gramians as in (2.25). This is possible because the Gramians are positive definite. This conclusion can be drawn directly by the inspection of their definitions in (2.19) and (2.22). For the singular matrix decomposition, the matrix U and V are orthogonal and Σ is a diagonal matrix with positive entries.

$$L_o^T L_r = U\Sigma V^T \tag{2.25}$$

Hence, T can be chosen as in (2.26) and, therefore, its inverse is given by (2.27).

$$T = L_r V \Sigma^{-1/2} \tag{2.26}$$

$$T^{-1} = \Sigma^{-1/2} U^T L_o^T \tag{2.27}$$

Using this change of variables produce Gramians that are both equal to Σ (This can be verified by direct substitution). The values of the diagonal can be used to measure how well reachable and observable a state is. Therefore, after changing variables, the new matrices can be truncated in a given position and used as a reduced model.

It is then necessary to chose the point to truncate the system. It can be done by taking the truncation index p as the smallest integer that satisfies (2.28). In this equations, δ is a quality factor normally close to the unity.

$$\frac{\sum_{i=1}^{p} \sigma_i}{\sum_{i=1}^{n} \sigma_i} \geq \delta \tag{2.28}$$

A very important theoretical result linked to balanced realization is the simple error bound that it satisfies (Enns, 1984). Its expression is given by (2.29). In this equation σ_k is the k-th singular value of the diagonal Gramians. Notice that in this expression only the neglected singular values contribute to the error of the system.

$$\|H(s) - H_r(s)\|_\infty < 2 \sum_{k=p+1}^{n} \sigma_k \tag{2.29}$$

There is however one missing point in the reduction process. One must compute the Gramians for the system. Using directly (2.22) and (2.19) is generally a bad idea. A practical computation of the cholesky factors of the gramians can be done if the time t is chosen to be infinite. Then the Gramians will be the solution of the two Lyapunov equations (2.30) and (2.31). A numerical strategy to solve these equations can be found in (Bartels and Stewart, 1972).

$$A^T W_o + W_o A + C^T C = 0 \tag{2.30}$$

$$A W_r + W_r A^T + B B^T = 0 \tag{2.31}$$

Balanced Truncation has several advantages as a reduction method. Is preserves the system stability (Antoulas, 2005), has an easily computable error bound and in practice has been shown to give excellent results. So why not use this technique for every system ? The bottleneck of balanced truncation is its computational cost due to the solution of the Lyapunov equations to obtain the Gramians. The cost is $O(n^3)$ making its application not practical for really high dimensional systems. Another downside is that it requires direct access to the system matrices. Therefore, if the system is only known by means of matrix-vector products, this approach is not possible.

Example of Application

This subsection ends with an example of application of the balanced truncation algorithm. The system that we use as illustration is a cantilever beam. This means that one extremity is fixed and the other one is free. An illustration can be seen in Figure 2.1. The numeric model is obtained by a FEM formulation as derived in (Panzer et al., 2009).

The input for this system is a vertical force in its free extremity. The output is the displacement of this same point. This system is very versatile and can have its number of elements as well as its mechanical properties easily modified. It has, then, became a benchmark for different Model Order Reduction techniques.

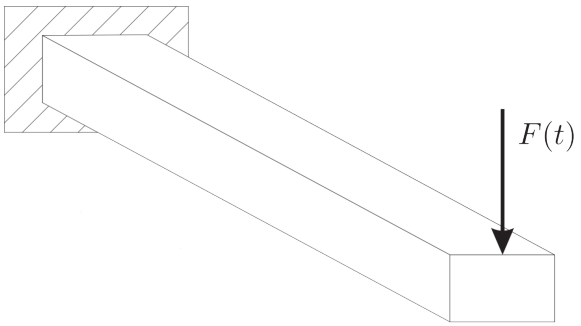


Figure 2.1: Illustration of the cantilever beam used as an example. Adapted from Panzer et al. (2009).

For our specific example, we have built a system of order 3600. In this theses we will encounter systems of much higher order. However, this will serve as a big enough number of equations to allow a fair comparison between the methods.

We have produced two reduced order models using the balanced truncation procedure. We have chosen 98 % and 99.9 % as the values of δ. This results in Reduced Order Models of order 6 and 10, respectively. Therefore, one can conclude that balanced truncation is a very powerful method to reduce the systems, considering that the original model has order 3600. Figure 2.2 shows how well the Reduced Order Models approximate the High-Fidelity One.

Having shown the frequency simulation of the systems, we also present the impulse response obtained by the High-Fidelity model and by the reduced ones in Figure 2.3.

It is hard to see the differences between the two reduced models. To solve this issue, Figure 2.4 shows the same signal but in a smaller time scale.

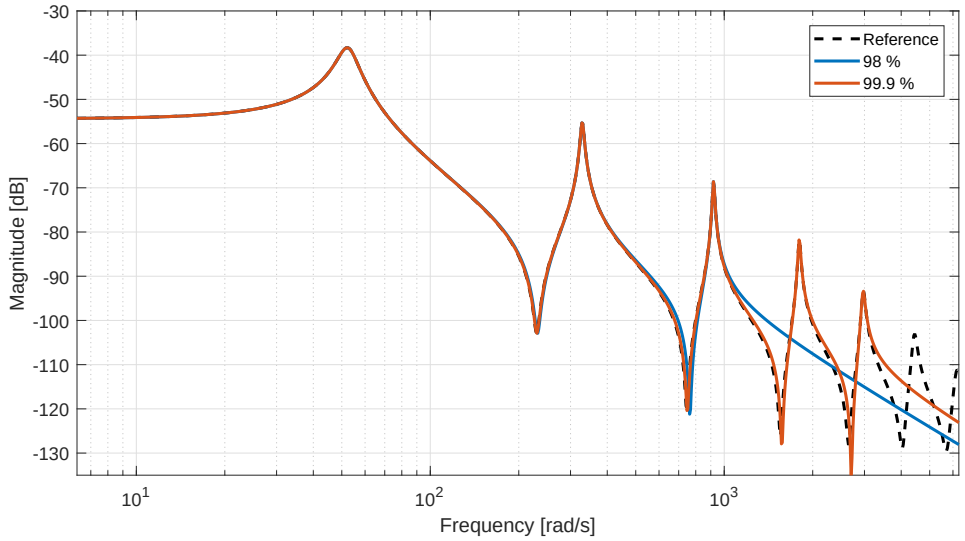


Figure 2.2: Simulation in the frequency domain of the reduced systems obtained by different values of δ

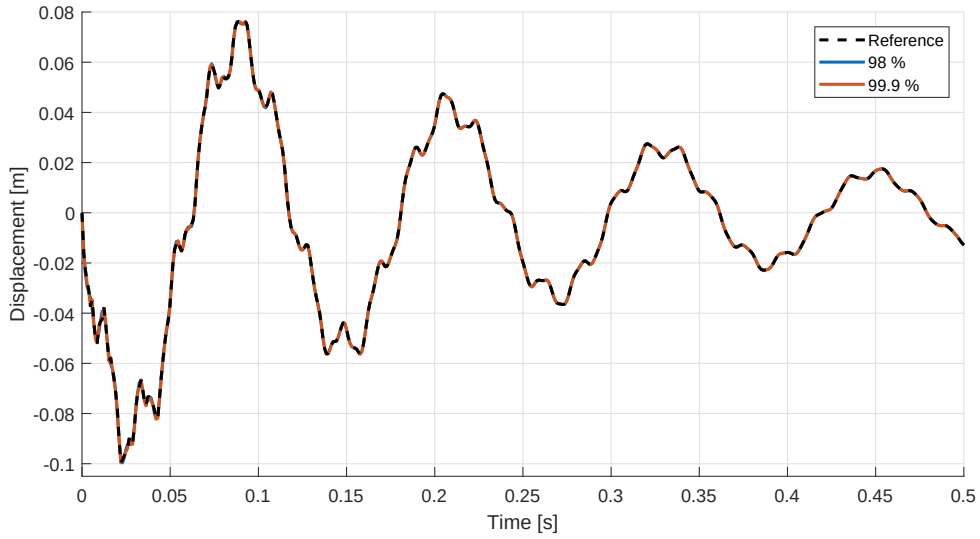


Figure 2.3: Comparison of the impulse response of the High-Fidelity Model and the Reduced Order Model

It is very important to note that the time domain simulation of the Reduced Order Models is only possible because balanced truncation preserves the stability during the reduction process. This is not true for other Model Order Reduction techniques and is one of the main advantages of this method.

Before proceeding to the next reduction technique, we present the time took for each

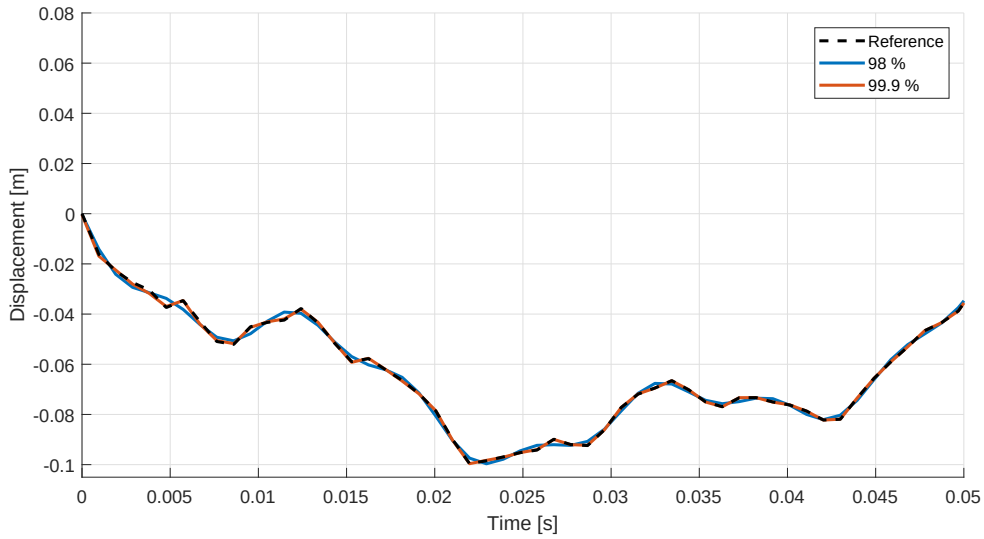


Figure 2.4: Zoom on the comparison of the impulse response of the High-Fidelity Model and the Reduced Order Model in the time domain.

of these process. For the High-Fidelity Model analysis in time and frequency took 125 s and 120 s, respectively. For the time domain analysis, the system has been discretized and the discrete time version has been used. Both procedures take only milliseconds when performed in the Reduced Oder Models. The reduction step takes 113 s, being 95 s for solving Lyapunov equations and 18 s to diagonalize the Gramians.

Therefore, we can conclude that the solution of the Lyapunov equations has been highly time consuming. This does not mean, however, that reducing the model is not worthy. A Reduced-Order Model can be computed once and used in the future to simulate many different inputs.

2.2.2 Proper Orthogonal Decomposition

Another reduction technique that has gained lots of popularity is the Proper Orthogonal Decomposition (Berkooz et al., 1993). Originally conceived as a method to approximate functions, it has been developed into a technique to reduce models based on simulations of the High-Fidelity Model. This method differentiates itself from other because due to its empirical approach it can be applied to non-linear system.

There are many ways of presenting this method. Some of them start with the general case of function approximation and use a lot of intermediate steps to arrive at model

order reduction. In this section, we have chosen to shine light on this method from the point of view of Model Order Reduction and taking into account that the simulation of a set of differential equations is solved by a quadrature method.

Classic Approach

As said before, one of the ideas of projection based model order reduction is to find a subspace spanned by $\mathcal{V}$ that can produce the best reconstruction of the original state space vector from the reduced variables. A direct way of doing this is to solve (2.1) and use the resulting vectors x in some sort as to build the columns of $\mathcal{V}$. This simulation is normally very time consuming and then can be done only in a small time interval or with steps of moderate size. A natural way to stock these vectors is at the columns of a matrix $S \in \mathbb{R}^{n \times s}$, called the snapshots matrix.

A technique is needed to find a small subspace that represents well the subspace spanned by the columns of S. Therefore, the columns of $\mathcal{V}$ will be picked as the vectors that maximize the restricted optimization problem in (2.32). One can recognize the direct application of a Lagrange multiplier to enforce the restriction. This problem can be motivated in the following manner: the alignment between two vectors is measured by the absolute value of the inner product between then. The first member of this equations looks for the vector v that maximize the sum of the squares of these inner products. In other words, it looks for the vectors that can better reconstruct the energy contained in S. Nevertheless, this value can be arbitrarily large because it increases without bounds as the modulus of v increases. To avoid this pathological case, the norm of this vector is fixed at unity by the second term of this equation.

$$v = \arg\min_{v} \mathcal{L} = \arg\min_{v} \|S^T v\|_F^2 - \lambda(v^T v - 1) \tag{2.32}$$

The maximum value of $\mathcal{L}$ is attained for the set of v's in the last line of (2.33). This development also shows how to deduce this condition.

$$\begin{aligned} \nabla_v(\|S^T v\|_F^2 - \lambda(v^T v - 1)) &= 0 \\ \nabla_v(v^T S S^T v) - \lambda \nabla_v(v^T v - 1) &= 0 \\ 2SS^T v - 2\lambda v &= 0 \\ SS^T v &= \lambda v \end{aligned} \tag{2.33}$$

The last line of this equation contains the restriction that the vectors v have to satisfy to maximize this optimization problem. For its importance, it is highlighted in (2.34). This equation represents a symmetric semi-positive definite eigenvalue problem. Therefore, it is known that an orthonormal eigenvector basis exists and that all the eigenvalues are nonnegative. These are exactly the kind of properties that are expected if one works with an energy formulation.

$$SS^T v = \lambda v \tag{2.34}$$

The matrix SS^T has dimension $n \times n$. Therefore, the computational burden to solve this eigenvalue problem is very high. There are two standard techniques to deal with this issue. It is here assumed that the number of sample points s is much smaller than the dimension of the problem n.

The first method consists in noticing that the matrix resulting from the product SS^T has at maximum rank s. This implies the nullity of $n - s$ eigenvalues. It is then only necessary to compute the s remaining eigenvalues. This can be done by looking at a different eigenvalue problem. If one multiplies (2.34) by S^T it will obtains (2.35).

$$S^T S S^T v = \lambda S^T v \tag{2.35}$$

Using the definition (2.36), it is possible to identify (2.35) as an eigenvalue problem.

$$u = S^T v \tag{2.36}$$

It is then necessary to obtain the vectors v already knowing the values for u. This can be done by multiplying (2.36) by S and using the definition of an eigenvalue to obtain (2.37).

$$v = \frac{1}{\lambda} S u \tag{2.37}$$

It is tempting to use these vectors as the solution. Nevertheless, these vectors are not normal and therefore do not respect the POD restriction imposed in (2.32). Solving this issue is very straight forward. It is only necessary to divide them by the norm of each vector. The result is given by (2.38).

$$\hat{v} = \sqrt{\frac{1}{\lambda}} S u \tag{2.38}$$

In this fashion, one can obtain all the eigenvectors that correspond to non-zero eigenvalues. It is natural to wonder what are the values for the other eigenvectors. However, these vectors are not necessary because they don't add any new information to the Reduced Order Model. This can be easily seen by computing the square of the norm of $S^T v$ for each of the eigenvectors. This is shown in (2.39).

$$\|S^T v\|_F^2 = v^T S S^T v = \lambda v^T v = \lambda \tag{2.39}$$

This analysis shows that the "contribution" of each vector to the reconstruction of the sampling matrix is proportional to its eigenvalue. Therefore, not knowing the eigenvector for zero eigenvalues is not an issue.

Singular Value Decomposition method

In this subsection, we present an alternative way to the solution of (2.35). It uses the Singular Value Decomposition (SVD) of the S matrix. It is beyond of the scope of this thesis to explain this decomposition in details. However, it is covered in a variety of linear algebra books as, for example, the excellent (Trefethen and Bau III, 1997).

Assume that the singular value decomposition of S has been computed and it is given by (2.40). In this equations, U and V are orthogonal matrices and Σ is a diagonal matrix. Notice that they are, most of the time, not squared. Some call this the reduced or economic singular value decomposition.

$$S = U\Sigma V^T \tag{2.40}$$

The idea of the SVD method consists in analyzing the product SS^T using this decomposition. This operation is shown in (2.41).

$$SS^T = U\Sigma V(U\Sigma V)^T = U\Sigma\Sigma^T U^T \tag{2.41}$$

This result can be interpreted as the diagonalization of the product SS^T. The matrix U contains the eigenvectors and $\Sigma\Sigma^T$ is a diagonal matrix containing the eigenvalues of the product. Therefore, the eigenvalues of this product are given by the square of the singular values of S.

Using this interpretation all the conclusions drew in the last section may be applied. This method, however, has the advantage of never actually computing the matrix product SS^T and therefore, the condition number of the problem is never squared.

Example of Application

As an example of application of the Proper Orthogonal Decomposition, the same cantilever beam used to illustrate balanced truncation is employed. As the first step in the POD process is the simulation of the High-Fidelity Model, we have applied a Pseudorandom Binary Sequence (PRBS) to the system and performed the simulation for 0.45 s with a time step of 3×10^{-4} s. Figure 2.5 shows this process. The simulation was performed using a discretization schema described in (Chen, 1995) and it took 82 s.

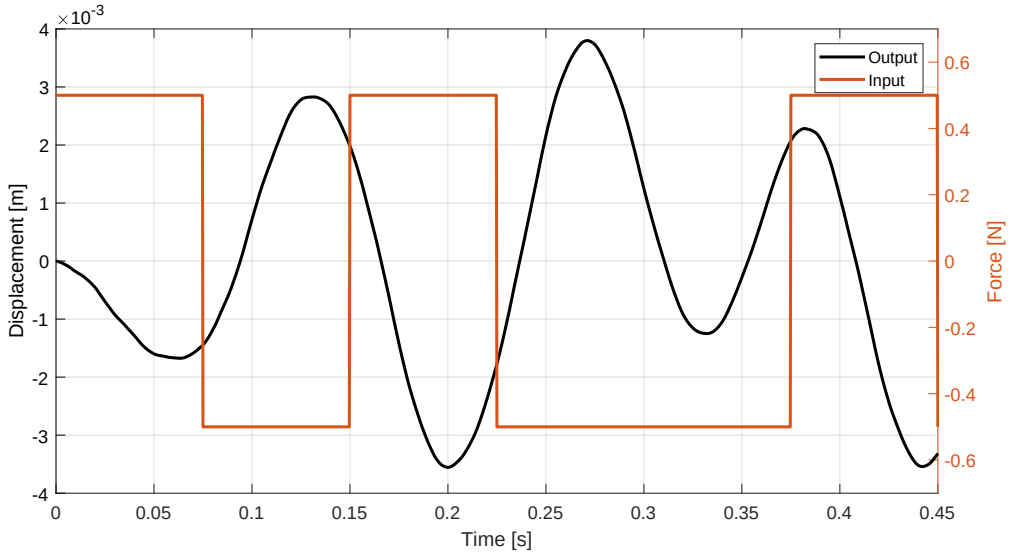


Figure 2.5: Sampling of the High-Fidelity System

To show the effect of the number of base vectors used to build the projection subspace, we have produced two Reduced Order Models with 20 and 40 vectors. The frequency response for these systems is shown in Figure 2.6. The reduction step took 2 s and the simulation time for the Reduced Order Model is negligible.

For this method we do not present a temporal simulation of the system. The reason for this is that the reduced order models obtained are not stable. This is one of the main disadvantages of the POD approach. However, in the next sections, we are going to introduce a method capable of solving this issue for any projection based Model Order Reduction technique.

Before presenting the next reduction technique, we note that the quality of the model can be changed by changing the number of vectors in the base. A completely similar

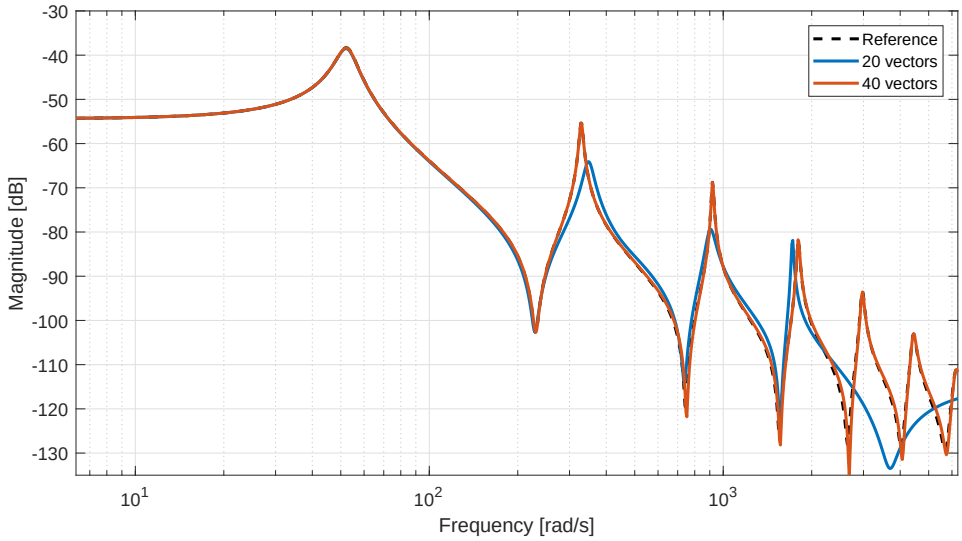


Figure 2.6: Frequency response for different number of vectors in the POD basis

situation is present in the balanced truncation approach. In these two methods, the time consuming phase of the reduction process is independent of the final order of the model. In the Proper Orthogonal Decomposition the reduction time is dominated by the simulation time. Therefore, the result we present is one instance of one specific input signal. Different results would be achieved if different inputs were used.

2.2.3 Moment Matching

Among the three presented reduction methods, this one will be treated in more detail. There is a simple reason for this : it is the method of choice for most of the applications in this thesis. In its classical form it has many problems that will become explicit in the development of this section. One of the main contributions of this work is various solutions for these limitations.

Moments of a Transfer Function

In a complementary fashion to the POD method, that uses time domain data, the moment matching technique uses frequency domain data to approximate the High-Fidelity Model. It is then expected to start by applying the Laplace transformation to (2.1) and assume a zero initial condition, to obtain the already presented matrix H that

contains the relations between inputs and outputs for the system. This matrix is shown in a slight different way as before in (2.42). In this equation $s \in \mathbb{C}$ and $A - sE$ is assumed to have an inverse except for a finite number of different values of s.

$$H(s) = -C(A - sE)^{-1}B \tag{2.42}$$

There are two different ways to obtain the moments (scalar multiples of the Taylor coefficients) for (2.42). The first one that we present is more intuitive but do not proves the general case. It is, nevertheless, a very good starting point. This exposition is based on (Nguyen, 2012). It will use the matrix version of the sum of the geometric series. Its expression is given by (2.43).

$$(\mathcal{I} - A)^{-1} = A^0 + A^1 + A^2 + A^3 + \dots \tag{2.43}$$

Factoring A from the matrix pencil and applying this equation to (2.42) results in (2.44).

$$H(s) = -C\left[\sum_{k=0}^{\infty} A^{-1}Es\right]^k A^{-1}B \tag{2.44}$$

This expression can be identified as the Taylor expansion of the transfer function if the expansion point is chosen as being zero. In order to obtain the answer for an arbitrary expansion point (s_0) one can rewrite (2.42) in a slight different, but equivalent, way, given by (2.45). Notice that the same quantity was added and subtracted from this expression leaving it unchanged.

$$H(s) = -C(A - s_0E - (s - s_0)E)^{-1}B \tag{2.45}$$

Identifying the terms in (2.45), the formula (2.44) can be used if the following substitutions are made : $A \rightarrow A - s_0E$ and $s \rightarrow s - s_0$. The resulting transfer function is then given by (2.46).

$$H(s) = -\sum_{k=0}^{\infty} C[(A - s_0E)^{-1}E]^k(A - s_0E)^{-1}B(s - s_0)^k \tag{2.46}$$

The k-th moment of a transfer function around an expansion point s_0 is defined as given by (2.47).

$$M_k(s_0) = C[(A - s_0E)^{-1}E]^k(A - s_0E)^{-1}B \tag{2.47}$$

The use of this definition allows the writing of (2.46) in a much simpler way given by (2.48).

$$H(s) = -\sum_{k=0}^{\infty} M_k(s_0)(s - s_0)^k \tag{2.48}$$

The problem with this first presentation is that we assumed in an intermediate step that the inverse of A exists. For the model order reduction framework, the current hypothesis is valid only for the matrix pencil $A - s_0E$.

Therefore, we present a method that is less intuitive but that does not use this assumption. It uses the direct Taylor expansion of a matrix function. To keep the discussion as simple as possible, a widely known formula is recalled in (2.49). The quantity F is assumed to be a matrix whose entries vary smoothly in function of the parameter t, and t_0 is an arbitrary expansion point.

$$F(t) = F(t_0) + F'(t_0)(t - t_0) + \frac{F''(t_0)}{2!}(t - t_0)^2 + \ldots \frac{F^{(k)}(t_0)}{k!}(t - t_0)^k + \ldots \tag{2.49}$$

It is then necessary to compute the derivative of the matrix $(A - sE)^{-1}$. This is not a problem since one can use the well known relation for the derivative of an inverse matrix as shown in (2.50). Its deduction is found as an exercise in (Golub and Van Loan, 2012).

$$\frac{dM^{-1}(t)}{dt} = -M^{-1}(t)\frac{dM(t)}{dt}M^{-1}(t) \tag{2.50}$$

Applying this equation many times to the matrix pencil $A - sE$ at the point s_0 it is possible to obtain a general formula for its k-th derivative as in (2.51).

$$\frac{d^k(A - sE)^{-1}}{ds^k} = k![(A - sE)^{-1}E]^k(A - sE)^{-1} \tag{2.51}$$

Plugging this expression in (2.49), one can deduce a general term for the moments of the transfer function as in (2.47).

Moment Matching

The main idea of a moment matching technique is to produce ROMs whose moments are the same of those of the HFMs for some given expansion points up to a given order. In this subsection, we show how one can do this without having the numerical problems that direct computation of moments leads to.

To simplify the notation in further discussion, the matrix pencil $A - s_0E$ will therefore be written as A_{s_0}. The first moment of a transfer function is given by (2.52).

$$M_0 = CA_{s_0}^{-1}B \tag{2.52}$$

For the projected system it given by (2.53). This expression was obtained by substituting the original matrices by the reduced ones.

$$\widetilde{M_0}(s_0) = C\mathcal{V}(\mathcal{W}^T A_{s_0}\mathcal{V})^{-1}\mathcal{W}^T B \tag{2.53}$$

Suppose that $A_{s_0}^{-1}B$ is in the column space of $\mathcal{V}$. Mathematically this is represented by the existence condition in (2.54).

$$\exists\, r \in \mathbb{R}^n \text{ such that } \mathcal{V}r = A_{s_0}^{-1}B \tag{2.54}$$

Substituting (2.54) into (2.53), one can conclude that the first moment of the original transfer function equals that of the reduced one, as shown in (2.55).

$$\widetilde{M_0}(s_0) = C\mathcal{V}(\mathcal{W}^T A_{s_0}\mathcal{V})^{-1}\mathcal{W}^T A_{s_0}\mathcal{V}r = C\mathcal{V}r = CA_{s_0}^{-1}B = M_0(s_0) \tag{2.55}$$

The same idea can be used to show that a clever choice of the columns span of $\mathcal{V}$ will match the second moment. The expression for the second moment of a reduced model is given by (2.56).

$$\widetilde{M_1}(s_0) = C\mathcal{V}[(\mathcal{W}^T A_{s_0}\mathcal{V})^{-1}\mathcal{W}^T E\mathcal{V}](W^T A_{s_0}\mathcal{V})^{-1}\mathcal{W}^T B \tag{2.56}$$

Condition (2.54) immediately simplifies this equation to (2.57).

$$\widetilde{M_1}(s_0) = C\mathcal{V}(\mathcal{W}^T A_{s_0}\mathcal{V})^{-1}\mathcal{W}^T E A_{s_0}^{-1}B \tag{2.57}$$

At this point a new condition in the columns span of $\mathcal{V}$ is introduced by (2.58).

$$\exists\, q \in \mathbb{R}^n \text{ such that } \mathcal{V}q = A_{s_0}^{-1}EA_{s_0}^{-1}B \tag{2.58}$$

Using this new condition in (2.57), immediately shows that both moments are equal, as shown in (2.59).

$$\widetilde{M_1}(s_0) = C\mathcal{V}q = CA_{s_0}^{-1}EA_{s_0}^{-1}B = M_1(s_0) \tag{2.59}$$

This argument can carry on to produce a general condition for the matching of the i-th moment, given that all the other lower moments are already matched. This condition is shown in (2.60).

$$\exists\, v \in \mathbb{R}^n \text{ such that } \mathcal{V}v = (A_{s_0}^{-1}E)^i A_{s_0}^{-1}B \tag{2.60}$$

Having this in mind, one can therefore define a matrix and a vector given, respectively, by (2.61) and (2.62). And use these entities to produce the subspace that will guarantees moment matching. For convenience, they will be named generating matrix and generating vector of the subspace.

$$K = A_{s_0}^{-1} E \tag{2.61}$$

$$v = A_{s_0}^{-1} B \tag{2.62}$$

This subspace can be written in the form of a Krylov subspace denoted by $\mathcal{K}(K, v)$. Its definition can be found in (2.63).

$$\mathcal{K}(K, v) = \text{span}\{v, Kv, K^2v, K^3v, \ldots\} \tag{2.63}$$

An immediate approach to the computation of the $\mathcal{V}$ matrix is to chose its i-th column as being $K^i v$. However, in practice this will prove to be a very poor choice. The vectors will very quickly become linearly dependent (only for a numerical point of view). Therefore, some caution has to be taken in the numerical computing of this subspace. Other than that, other questions may arise. What if this subspace is complex ? How should it be computed for different expansion points ? How to chose the expansion points ? How to guarantee that the reduced system is stable ?

These questions are the target of the next chapter. In the following sections, however, we will present the classic Arnoldi Process which consists in a robust way of computing the basis for the Krylov subspace.

Simple Arnoldi Process

As seen in the last section, one needs to compute the subspace spanned by (2.63) avoiding as much as possible the numerical errors in the process. To start with a simple problem, only the two first vectors of the naive approach are drawn in Figure 2.7.

Although this picture is in two dimensions, it must be thought as two vectors lying in an n-dimensional space.

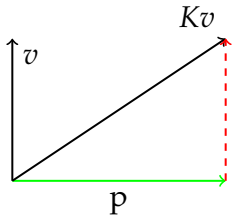


Figure 2.7: Depiction of the first two vectors of the subspace computation

Vectors v and Kv are linearly independent and therefore can be used as a basis for a two dimensional space in $\mathbb{R}^n$. Numerically, however, it is much better to use an orthonormal base that spans the same subspace. This can be done by subtracting the projection of Kv into v from the original Kv followed by a normalization step. This process is known as Gram-Schmidt orthogonalization (Strang, 2011).

It is very useful to create an intuitive understanding of whats orthogonalization means in the framework of Model Order Reduction. The subspace spanned by the columns of $\mathcal{V}$ is the important quantity and not the vectors that are used to represent it. Vectors v and Kv are linearly independent but the second one has non zero projection into the first one. This means that some of the information contained in Kv is not new. It was already present in v. Using p instead of Kv does not change the spanned subspace but, as it is orthogonal to v, the direction in the space that it points is completely new. This remark leads to the conclusion that directions of the space can be interpreted as information. The idea behind the computation of the matrix $\mathcal{V}$ is then to keep only the new information brought by each new vector.

This idea leads to Anoldi Iteration (Arnoldi, 1951). Each new column of the matrix is orthogonalized in relation to all others. At this point it is important to notice that this process can be applied even if only the product matrix-vector is known, represented by a black box process in Figure 2.8. This is normally the only information that can be obtained if matrix compression techniques are being used.

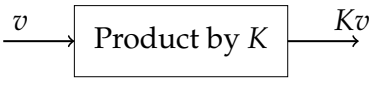


Figure 2.8: Black box model for the matrix product

This section deals with the deduction of the Arnoldi iteration and presents a brief overview of its properties. This is a classical topic that will be presented anyway due to its importance for the subject. The reader can consult (Trefethen and Bau III, 1997) (Golub and Van Loan, 2012) or (Recipes, 2007) and references therein for further information.

Suppose that the process of orthogonalization is in its j+1 iteration and that already mentioned process of multiplication by K and orthogonalization has already been done to j vectors. There is then a matrix $\mathcal{V} \in \mathbb{R}^{n \times j}$ that stores each of the produced vectors as its columns.

To obtain the next vector, one multiplies the last column by K and perform the orthogonalization process. This is shown in (2.64). The resulting vector r_{j+1} is not directly called v_{j+1} because it is not yet unitary.

$$r_{j+1} = Kv_j - \sum_{k=1}^{j} \text{proj}_{v_k} Kv_j \tag{2.64}$$

This equation can be further developed by plunging the expression for the projection of a vector. In the development, there is no denominator because the fact that the vectors v are unitary is used.

$$Kv_j = r_{j+1} + \sum_{k=1}^{j} (v_k^T Kv_j) v_k \tag{2.65}$$

In order to obtain a more concise notation, one can define a matrix H (it should not be confused with the transfer matrix of a linear system) whose entries are given by (2.66). This matrix is an example of an upper Hessenberg matrix. This means that in its lower triangular part, only the second diagonal in not zero.

$$H_{lm} = \begin{cases} v_l^T Kv_m & \text{if} \quad l \leq m \\ \|r_{m+1}\| & \text{if} \quad l = m+1 \\ 0 & \text{otherwise} \end{cases} \tag{2.66}$$

Using this definition, (2.65) can be written as (2.67). It is important to notice that the upper summation limit has changed to include the term r_{j+1}.

$$Kv_j = \sum_{k=1}^{j+1} v_k H_{kj} \tag{2.67}$$

This summation can be put in a very useful matrix notation by defining some derived matrices. The first two ones are $\mathcal{V}_j$ and $\mathcal{V}_{j+1}$. The first one is a matrix containing the first j computed vectors. By analogy, $\mathcal{V}_{j+1}$ contains the first $j+1$ vectors. In the same fashion, two other matrices are defined : H_j and H_{j+1}. The second one is a $j+1 \times j$ matrix, as in (2.67). The elimination of the last line of this matrix results in H_j, a square matrix. Using these new derived matrices, (2.67) can be written in pure matrix notation as done in (2.68).

$$K\mathcal{V}_j = \mathcal{V}_{j+1} H_{j+1} \tag{2.68}$$

The right hand side of this expression can be written as the sum of two other ones as in (2.69). The vector e_j, as usual, contains zero in all its positions except in its j-th one, where it has a one.

$$\mathcal{V}_{j+1} H_{j+1} = \mathcal{V}_j H_j + v_{j+1} \|r_{j+1}\| e_j^T \tag{2.69}$$

This last step can appear very complex, but a simple illustration for a small case can improve a lot its comprehension. Figure 2.9 shows this for a 4×3 matrix.

Substituting (2.69) into (2.68), allows the writing of the Arnoldi iteration as (2.70).

$$K\mathcal{V}_j = \mathcal{V}_j H_j + \|r_{j+1}\| v_{j+1} e_j^T \tag{2.70}$$

$$\left[\begin{array}{c|c|c|c} v_1 & v_2 & v_3 & v_4 \end{array}\right]\left[\begin{array}{ccc} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ 0 & H_{32} & H_{33} \\ \hline 0 & 0 & \|r_4\| \end{array}\right] = \left[\begin{array}{c|c|c} v_1 & v_2 & v_3 \end{array}\right]\left[\begin{array}{ccc} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ 0 & H_{32} & H_{33} \end{array}\right] + \left[\begin{array}{c|c|c} 0 & 0 & \|r_4\|v_4 \end{array}\right]$$

Figure 2.9: Example of the separation in (2.69) for a small matrix.

By the multiplication of (2.70) by the left by $\mathcal{V}_j^T$, the decomposition in (2.71) can be obtained. In this step, the orthogonality property was used.

$$H = \mathcal{V}^T K \mathcal{V} \tag{2.71}$$

At this point, it is important to point out that if the matrix K is symmetric, then H will clearly also be symmetric. But, H is also a Hessenberg matrix. Therefore, for this special case, H will be a tridiagonal matrix and the iteration can be further simplified. This special case is known as Lanczos iteration (Lanczos, 1950).

In order to perform a robust computer implementation, two changes are done in the pure mathematical process. The first and most obvious one is to use modified Gram-Smidth instead of the classic version of the algorithm. The second one is dealing with the stop criteria used. In practice the user will ask for only a limited number of columns in $\mathcal{V}$ depending of how many moments are to be matched. However, as computers work in finite precision, sometimes the produced vector may be considered to be linear dependent to those already in the basis. This is called deflation. This situation must be detected and the process should stop at this point. Algorithm 1 shows how to implement the Arnoldi process having these two points in mind.

In the specific case of model order reduction, only the projection matrix $\mathcal{V}$ is of importance. Therefore, one does not need to store the matrix H as done in this example. There is, then, room for improvement in this algorithm if the only objective of its implementation is to perform Model Order Reduction. This is, however, such an important building block for the moment matching process, that we have chosen to show it in its original form. These improvements are left to the next algorithm presented in the following chapter showing how to compute the projection matrix for many expansion points.

Algorithm 1 Numerical computation of the Krylov Subspace

procedure V = ARNOLDI ITERATION(K, v, maxIter, ϵ)

  $V(:,1) \leftarrow \frac{v}{\|v\|}$
  iter $\leftarrow 1$

  while iter $\leq$ maxIter **do**

    r $\leftarrow KV(:, end)$ ▷ This product can be obtained by a black box process
    normR1 $\leftarrow \|r\|$
    for k = 1:numCols(V) **do**
      $H(k,j) \leftarrow \frac{V(:,k)^T r}{V(:,k)^T V(:,k)}$
      $r \leftarrow r - H(k,j)V(:,k)$
    end for

    normR2 $\leftarrow \|r\|$
    $H(j+1,j) \leftarrow$ normR2
    if $\frac{normR2}{normR1} \leq \epsilon$ **then** ▷ Dealing with deflation
      return V
    end if

    iter $\leftarrow$ iter + 1
    $V(:,iter) \leftarrow \frac{r}{normR2}$

  end while

  return V
end procedure

Extension to Discrete Time Systems

It can be useful to apply moment matching to models that have been discretized. As an example, in the following chapters we will present a model that has been derived using Finite Differences in Time Domain to compute electromagnetic fields and it is naturally discrete (Elsherbeni and Demir, 2016).

The extension to discrete time system can be done in a very straight forward manner. Every property holds exactly as shown, but the variable s has to be replaced by $e^{j\Omega}$, where Ω is the normalized frequency ranging from $-\pi$ to π.

To understand this, one has to consider the eigenfunctions for discrete time systems. As the time passes and the initial state solution fades away, a stable system excited by a

function of the form α^k outputs a signal of the form $K(\alpha)\alpha^k$, where K and α are complex numbers.

To form a sinusoidal function, one has to combine complex exponentials of the form e^{jwk}. This can be achieved by selecting $\alpha = e^{jw}$. Therefore, the transfer function has the form $H(e^{jw})$, which justifies the proposed change of variables.

Example of Application

As an example of application of the Moment Matching, we use the same problem used to illustrate balanced truncation and proper orthogonal decomposition. Only one point of expansion was chosen at 0 Hz. This will not be the case in other applications of this method, but we have chose this to simplify the result. The two produced Reduced Order Models have order 10 and 20. The frequency simulation of these systems is shown in Figure 2.10.

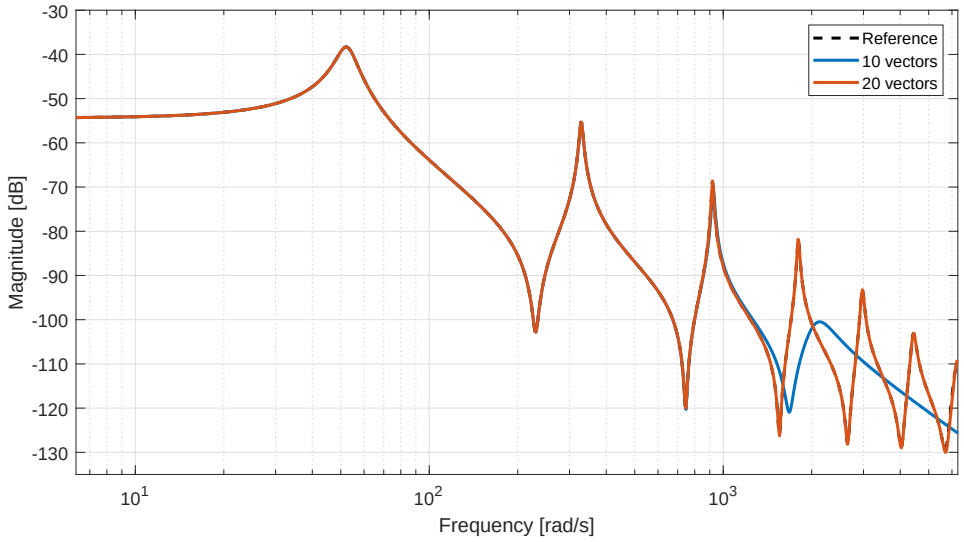


Figure 2.10: Frequency response for different number of moments matched

For this method we also do not present a temporal simulation of the system because the models are not stable. This is one of the subjects of the next chapter based on improving the Moment Matching.

At this point we are able to compare the time took for each one of the reduction methods that have been presented. For moment matching, the reduction steep took 0.8 s and 1.0 s for 10 and 20 vectors, respectively.

If we choose the best model obtained by each method and consider only the frequency simulation (some of them are not stable), we can summarize the time used by each method as in Table 2.1.

Method	Reduction Time [s]	Simulation Time [s]	Order	Stable
Original Model	–	129	3600	yes
Balanced Truncation	113	< 0.1	10	yes
POD	84	< 0.1	40	no
Moment Matching	1.0	< 0.1	20	no

Table 2.1: Time comparison between the different methods.

2.2.4 Circuits and special cases

The methods presented in this chapter are completely generic and do not assume any property of the matrices A or E. This generality comes at a price. Some results were not stable and consequently not passive.

There are systems, however, that have very well understood properties which can be exploited to avoid such problems. The example of interest in this section will be circuits. The literature contains special algorithms to deal with them with results that are far superior to the ones that can be obtained by blindly applying generic methods. Special attention must be payed to the PRIMA and SPRIM algorithms (Odabasioglu et al., 1997) (Freund, 2011b).

For general circuit equations, one has the properties $A \preceq 0$ and $E \succeq 0$. Knowing that the poles of the system are a subset of its generalized eigenvalues and that their real parts are given by (2.72), it can be concluded that orthogonal projections will produce stable Reduced Order Models.

$$\mathfrak{R}\{\lambda\} = \frac{1}{2}\frac{u^*(A + A^T)u}{u^*Eu} \tag{2.72}$$

2.3 Conclusion

In this chapter we have presented different and well established MOR methods. The resulting systems are good approximations of the original model that can be used to approximate the simulation of the original model.

These different methods have different properties, applications and caveats. The balanced truncation approach produces stable ROMs and has rigorous error bounds.

Moment Matching and Proper Orthogonal Decomposition lack a formal error bound. However, they are cheaper and the second one is known to work for non linear systems. As we were able to see in this chapter, Moment Matching is very fast when compared to the other methods.

There are still open questions that should be answered before Moment Matching can be used as a general method for MOR. One must consider the problem of stability and passivity as well as the accuracy of the models.

The next chapter will address these questions and present how to assure the stability and precision of the ROMs.

Chapter 3

Improving Moment Matching: Accuracy, Stabilization and Time Simulation

3.1 Introduction

The previous chapter presented some well known Model Order Reduction methods. Some of their limitations were also explained. This chapter aims at overcoming the main disadvantages of the Moment Matching approach in order to allow its use in our systems.

The three main objectives of the next sections are to obtain real, stable and accurate reduced order models. In the case of impedance probing, the systems should also be passive. Each one of these objects leads to a specific topic that is crucial to practical application of Reduced Order Models.

Real systems are necessary to perform dynamic simulation and facilitate meaningful physical interpretations. Imagine that the cantilever bean presented in the last chapter has been reduced. We would not be able to assign meaning to displacements that were predicted to be complex numbers.

Stability is also an essential property that must be preserved to perform time domain simulation. If we assume that the dynamic system can be diagonalized, then all its modes are independent and correspond to complex exponentials. If the argument of any of these functions has real part bigger than zero, given enough time the solution will reach arbitrarily large values. In theory it is possible to produce initial conditions and inputs that do not interact with unstable modes. However, in practice due to noise

in the signals or to numerical errors, the simulations will always diverge. Therefore, time simulation of unstable systems are to be considered meaningless.

One of the main advantages of reduced order models is that once obtained, they can be used instead of the High Fidelity Model. Therefore, they should be fast and accurate. Their speed is guaranteed by the reduced number of equations. The accuracy, however, will only be enforced locally by the Moment Matching approach. One must produce a method that ensures the accuracy over a large frequency range.

Before dealing with each of these topics, we present a classic method to reduce numerical problems when many expansion points have to be placed in a large frequency range. Although this is a somehow classic procedure, it will be necessary to understand one of our contributions.

3.2 Placing Many Expansion Points

In the last chapter a condition for the moments to be matched was deduced. In theory it is possible to well represent the transfer function by only choosing one expansion point and matching many moments.

In practice, however, this may pose several numeric issues. Intuitively, one can think of this process as trying to obtain global information about a function only having access to its local information (at the expansion point). We know by the Taylor series theory that this is possible in principle. If the Taylor series is convergent, one can infer information about the function far away from the expansion point by having knowledge of its derivatives. In the Moment Matching approach, this would require the computation of basis vectors that are very similar. If the orthogonalization process is not extremely accurate (which is often the case), it is very difficult to extract new information from these vectors.

3.2.1 Block Arnoldi Process

One solution to this problem is to place several expansion points with only a few moments matched at each point. Coming back to our Taylor series analogy, this would mean choosing an expansion point closer to the value that we would like to compute. Suppose that one must calculate the value of $\sin(89^\circ)$. It is perfectly fine to chose zero

as the expansion point and perform the series summation up to a large power before truncating. However, it would be much clever to simply chose ninety as the expansion point and only compute two or three terms.

To match moments at different expansions points, it is then necessary to build the matrix $\mathcal{V}$, whose span is the union of the spans for the individual matrices $\mathcal{V}_i$ obtained for each expansion point. Mathematically we would require (3.1) to be satisfied. Therefore, the condition in (2.60) would always be satisfied.

$$\text{span}\{\mathcal{V}\} = \bigcup_i \text{span}\{\mathcal{V}_i\} \tag{3.1}$$

A matrix $\mathcal{V}$ that satisfies this constraint can be directly constructed by the concatenations of the individual matrices of each point. This process is shown in (3.2).

$$\mathcal{V} = [\mathcal{V}_0 \mid \mathcal{V}_1 \mid \ldots \mid \mathcal{V}_k] \tag{3.2}$$

There is, however, a downside in doing this directly. This matrix is not orthogonal and is not guarantee to have linear independent columns. Some of the subspaces can have redundant information. This is coherent with our previously interpretation that directions is the subspace translate directly into information.

To obtain an orthonormal basis of the subspace, one can utilize a QR or a SVD decomposition. This is, however, not a robust way to compute these subspaces. A more stable form of doing this is to modify the Arnoldi process to orthogonalize each newly computed vector in respect to all the previous subspaces. To make this idea more concrete, we present an example. Imagine that two expansion points are placed at some given frequencies. When computing the basis vectors of the subspace for the first point, each newly computed vector would be orthogonalized in respect to the already computed vectors for this point. When doing the computation for the vectors of the second expansion point each vector would be made orthogonal in respect to all the vectors of the first subspace and in respect to the ones already present in the

Algorithm 2 Numerical computation of the Krylov Subspace for many expansion points

procedure V = ARNOLDI ITERATION(K, $\mathcal{V}_0$, v, maxIter, ϵ)

  $V(:,1) \leftarrow \frac{v}{\|v\|}$
  iter $\leftarrow 1$

  while iter $\leq$ maxIter **do**
    r $\leftarrow KV(:,end)$ ▷ This product can be obtained by a black box process
    normR1 $\leftarrow \|r\|$

    for k = 1:numCols(V) **do**
      $r \leftarrow r - \frac{V(:,k)^T r}{V(:,k)^T V(:,k)} V(:,k)$
    end for

    for k = 1:numCols(V_0) **do**
      $r \leftarrow r - \frac{V_0(:,k)^T r}{V_0(:,k)^T V_0(:,k)} V_0(:,k)$
    end for
    normR2 $\leftarrow \|r\|$
    $H(j+1,j) \leftarrow$ normR2

    if $\frac{normR2}{normR1} \leq \epsilon$ **then** ▷ Dealing with deflation
      return V
    end if
    iter $\leftarrow$ iter + 1
    $V(:,iter) \leftarrow \frac{r}{normR2}$
  end while

  return V
end procedure

basis for the second subspace. In the end, the matrix $\mathcal{V}$ would already be orthogonal and would not need a rank-revealing process. Algorithm 2 shows how this can be implemented.

This algorithm is known in the literature as the Block Arnoldi Process (Bouhamidi et al., 2011). Each added vector is orthogonal to all the set of already computed ones. This guarantees that no redundant information is included in the new subspaces.

3.3 Real Subspace for Time Domain Simulation

In most of the practical applications, the value of s in the transfer function is chosen as being purely imaginary. This choice allows the interpretation of the transfer function as the action that the system performs into a purely sinusoidal signal. (Chen, 1995).

This will produce $\mathcal{V}$ matrices with complex entries. If only the analysis of the transfer function is envisaged, this is not a problem. However, it will not be possible to interpret the dynamic system behavior in time. The integration of this systems to external software may also be a problem.

Hence, one needs to find a new matrix $\mathcal{V}_r$ that is composed only of real entries and spans the same subspace as the complex one $\mathcal{V}$. The naive manner to do this is to recognize a complex vector as a linear combination of real vectors with complex coefficients. Equation (3.3) shown this interpretation.

$$z = \underbrace{1}_{\alpha_0} \overbrace{r}^{v_0} + \underbrace{j}_{\alpha_1} \overbrace{i}^{v_1} \tag{3.3}$$

This idea can be used to produce a real matrix $\mathcal{V}_r$ as (3.4) that contains the span of the original complex matrix (Freund, 2011a).

$$V_r = [\Re\{V\} \;\; \Im\{V\}] \tag{3.4}$$

In a first analysis one may think that this new matrix is twice as big as the original one. This is true in respect to the number of columns. However, in a computer, a complex number takes twice the memory of a real number. Therefore, this is not a true issue.

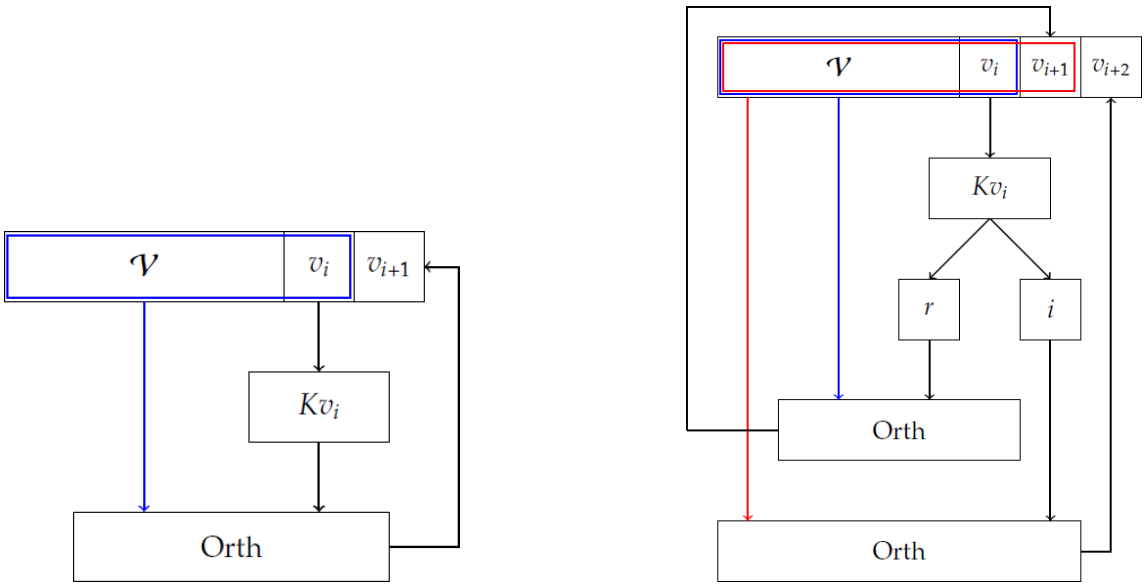
In practice, nevertheless, the direct use of this method presents poor numerical properties. Also, this new matrix is not guaranteed to have full column rank. Hence,

a rank-revealing process must be executed after the construction of the matrix. This problem is exactly the same that we presented in the last section. We have two matrices whose spans we would like to join.

If vectors are discarded during rank-revealing process, it means that during the Arnoldi process computational power was used to compute components that do not bring any new information to the subspace.

In order to obtain a numerically robust algorithm, we propose a modification to the Block Arnoldi algorithm in order to orthogonalize both the real and imaginary parts of the vectors during the construction process. We call this process Orthogonalization on the Fly for Moment Matching (OFMM). It will produce a real subspace that does not need a further orthogonalization step. This idea is closely related to the one presented by (Ruhe, 1994). However, we have found that to the application of Model Order Reduction via Moment Matching, the algorithm can be further simplified which reduces the computational cost and makes the implementations easier.

In order to well understand the new proposition, Figure 3.1 compares the original Arnoldi Process and the new one.



(a) Classic Approach (b) OFMM

Figure 3.1: Scheme comparing the Classical Arnoldi process and the proposed one

In the original Arnoldi Process, the last vector added to the subspace, denoted by v_i in Figure 3.1 (a), is multiplied by the generating matrix K and then orthogonalized to all the already computed vectors.

The proposed modification is shown in 3.1 (b). It starts in the same way of the original one by the multiplication of the last computed vector in the matrix $\mathcal{V}$ by the generating matrix K. Note that this vector is real, but when multiplied by a complex matrix K it results in a complex vector. It is then separated in real r and imaginary i parts. As a first step, the real part is orthogonalized to the all previously computed vectors and the resulting vector is incorporated into $\mathcal{V}$. At a second step, the imaginary part is orthogonalized to the entire matrix $\mathcal{V}$, including the recently added result from the real part. The new vector is then incorporated into the matrix $\mathcal{V}$.

It is natural to wonder if this new process will results in the same subspace spanned by the columns of the matrix in (3.4). The immediate answer to this question is no. In general, the computed subspaces are not the same. However, for the application of Model Order Reduction, the structure present in the generating matrix and starting vector produces the correct subspace. The proof of this fact is rather long and can be seen in the appendix B. The implementation can be seen in Algorithm 3.

Note that deflations (linear dependence of the newly produced vectors with respect to the subspace) has to be treated in a slightly different manner. When doing the product Kv_i one could find the real or imaginary parts to be linearly dependent to the subspace. Therefore, both these conditions have to be checked.

Having this algorithm in hand, one can produce real projection subspaces. This is one step in the direction of allowing time domain simulation of the Reduced Order Models. However, having real matrices is not the only prerequisite. One must have stable systems to allow this kind of simulation. Therefore, the next section treats this issue.

Algorithm 3 Arnoldi Iteration with on the fly orthogonalization

procedure V = ARNOLDI ITERATION(K, v, maxIter, ϵ)

  $V(:,1) \leftarrow \frac{v}{\|v\|}$
  iter $\leftarrow 1$

  while iter $\leq$ maxIter **do**
    q $\leftarrow KV(:,end)$ ▷ This product can be obtained by a black box process
    r $\leftarrow \Re\{q\}$ ▷ Real Part

    normR1 $\leftarrow \|r\|$
    for k = 1:numCols(V) **do**
      $r \leftarrow r - \frac{V(:,k)^T r}{V(:,k)^T V(:,k)} V(:,k)$
    end for
    normR2 $\leftarrow \|r\|$
    if $\frac{normR2}{normR1} > \epsilon$ **then**
      $V(:,end+1) \leftarrow \frac{r}{normR2}$
    end if

    i $\leftarrow \Im\{q\}$ ▷ Imaginary part
    normI1 $\leftarrow \|i\|$
    for k = 1:numCols(V) **do**
      $i \leftarrow i - \frac{V(:,k)^T r}{V(:,k)^T V(:,k)} V(:,k)$
    end for
    normI2 $\leftarrow \|i\|$
    if $\frac{normI2}{normI1} > \epsilon$ **then**
      $V(:,end+1) \leftarrow \frac{i}{normI2}$
    end if

    if $\frac{normI2}{normI1} \leq \epsilon$ and $\frac{normR2}{normR1} \leq \epsilon$ **then** ▷ Dealing with deflation
      return V
    end if
    iter $\leftarrow$ iter + 1
  end while

  return V
end procedure

3.4 Stability of Reduced Order Models

In this section a new method for stabilizing projection based model order reduction is presented. There are many proposed methods in the literature. however, all of them are, to the authors knowledge, limited. In (Bond and Daniel, 2008a) an optimization

problem has to be solved in order to find a matrix $\mathcal{W}$ that guarantees stability. This solution is often subject to numerical problems. Another approach can be found in (Amsallem and Farhat, 2012) where one must compute more projection vectors that will be used in order to find a subset of them that produce a stable model. This method is also limited to diagonal E matrices in (2.1). For the specific case of circuits, there are two very successful methods (one is the evolution of the other) that will guarantee stability (Odabasioglu et al., 1997) and (Freund, 2011b).

The method that has been developed in this work is the most general, for moment matching, that the author is aware of. It is based in the idea of doing a smart choice of $\mathcal{W}$ so that the reduced system present only stable poles. This will not change the matrix $\mathcal{V}$, keeping the moment matching property. This choice will not be based into an optimization process but in the perturbation of the eigenvalues of the reduced order model.

The first choice of the projection subspace is $\mathcal{W} = \mathcal{V}$. As known, there is no guarantee that all the poles of the projected system will be contained in the left half of the complex plane. The physical interpretation of this fact is that the even though the HFM is stable the ROM may have unstable behavior. The simulation of the dynamical behavior of the system is, then, impossible.

The poles of the reduced system are a subset of the eigenvalues given by the following generalized eigenvalue problem in (3.5).

$$A_r u = \lambda E_r u \tag{3.5}$$

In the following sections, we will demonstrate how to perform smart modifications in the projection matrix to assure that all the eigenvalues have negative real part.

3.4.1 Rank-1 Updates

The main idea of the developed method is to perform a rank-1 update in this eigenvalue problem in order to control the placement of the unstable poles. Equations (3.6) shows such update.

$$(A_r + E_r u v^*)u = \lambda_u (E_r + E_r u w^*)u \tag{3.6}$$

As demonstrated in the appendice A, this update preserves the values of all the eigenvalues of the system except to the one related to the eigenvector u. Its new value λ_u is related to the old one λ as in (3.7).

$$\lambda_u = \frac{\lambda + v^* u}{1 + w^* u} \tag{3.7}$$

This schema updates directly the reduced matrices A_r and E_r. However, in MOR techniques based on projections, the user only controls the subspaces in which the model will be projected. In the following paragraphs, it will be shown how to translate a rank-1 update of $\mathcal{W}$ into a rank-1 update of the reduced matrices.

The first choice of $\mathcal{W}$ is done as being equal to $\mathcal{V}$. It is very likely that this model will not be stable. A new choice of $\mathcal{W}$ can be done by doing a rank one update of the original matrix. This update is shown in (3.8). In this equation, $\mathcal{W}_0$ is the initial choice of $\mathcal{W}$. Not calling this $\mathcal{V}$ will allow to use any matrix as the starting point. This will be very useful due to the iterative nature of the method that is being presented. The vectors p and q are, for the moment, two arbitrary vectors whose sizes respects the constraints in matrix multiplication.

$$\mathcal{W} = \mathcal{W}_0 + pq^* \tag{3.8}$$

This update in the $\mathcal{W}$ matrix will produce a new generalized eigenvalue problem given by (3.9).

$$(\mathcal{W}_0 + pq^*)^T A \mathcal{V} u = \lambda_u (\mathcal{W}_0 + pq^*)^T E \mathcal{V} u \tag{3.9}$$

Doing the products and identifying the original reduced matrices will produce a more useful form given by (3.10).

$$(A_r + \underbrace{qp^*A\mathcal{V}}_{uv^*})u = \lambda_u(E_r + \underbrace{qp^*E\mathcal{V}}_{uw^*})u \tag{3.10}$$

This form of the equation allows the direct identification of the vectors v and w. These conditions are given by (3.11). The vector q must also be chosen equal to $E_r u$ to guarantee consistency.

$$v^* = p^*A\mathcal{V} \qquad\qquad w^* = p^*E\mathcal{V} \tag{3.11}$$

Using this condition in (3.7) will result in a formula for the eigenvalue of the ROM in function of the arbitrary vector p. This is given by (3.12).

$$\lambda_u = \frac{\lambda + p^*A\mathcal{V}u}{1 + p^*E\mathcal{V}u} \tag{3.12}$$

At this point, p is a degree of freedom of the problem. It must, however, not be chosen as a vector in the span of $\mathcal{W}_0$. If this was the case, then there would exist a vector m such that $p = \mathcal{W}_0 m$. And using this relation in (3.12) results in $\lambda_u = \lambda$.

Respecting this restriction, one can write p as a scalar times another arbitrary vector without any loss of generality. To avoid adding a new variable, p will now be considered as being αp. This Equation can now be solved by α which will result in (3.13).

$$\alpha^* = \frac{\lambda_u - \lambda}{p^*(A - \lambda_u E)\mathcal{V}u} \tag{3.13}$$

A sensible choice of λ_u is to pick the negative of λ. Using this in the general Equation (3.13) will produce a more specif version given by (3.14).

$$\alpha^* = -\frac{2\lambda}{p^*(A + \lambda E)\mathcal{V}u} \tag{3.14}$$

This equation can be used to compute the value of α that will move an unstable pole to the left half of the complex plane. If only real eigenvalues pose problems, the iterative use of this process will produce a stable ROM. There is however a problem with the following procedure to fix conjugate pairs of unstable eigenvalues. The use of this algorithm will produce matrices $\mathcal{W}$ with complex entries. Which, as discussed early, should be avoided.

Before preceding to the next subsection, we present the Algorithm 4 that summarizes this section, showing how to compute the Rank-1 update to stabilize the system.

Algorithm 4 Update to remove one real unstable eigenvalue

1: **procedure** $\mathcal{W}$ = Remove Unstable Eigenvalue($\mathcal{V}$, $\mathcal{W}$, A, E, E_r, u, λ)

2: $q \longleftarrow E_r u$
3: $p \longleftarrow$ random vector
4: $p \longleftarrow$ orthogonal vector($\mathcal{W}, p$)

5: $\alpha \longleftarrow \frac{2\lambda}{p^T E \mathcal{V} u \lambda - p^T A \mathcal{V} u}$

6: $p \longleftarrow \alpha p$
7: $\mathcal{W} = \mathcal{W} + pq^T$

8: **return** $\mathcal{W}$
9: **end procedure**

3.4.2 Rank-2 Updates

This first method is, nevertheless, not useless. It is perfectly applicable to remove unstable purely real poles of ROM and will be used as a first step in the complete stabilization schema. To deal with the case of complex conjugate eigenvalues, the

developed method uses the same principle of its real version. If a rank-1 update can move one pole, it is natural to assume that a rank-2 update can move two poles at once.

We begin by writing the eigenvalue problem for the two eigenvectors and eigenvalues that are to be moved. This can be found in (3.15). In this equation U is the matrix that contains the two eigenvectors and D is a diagonal matrix with their corresponding eigenvalues.

$$A_r U = E_r U D. \tag{3.15}$$

A rank-2 update of this problem can be done in an analogous way to (3.6). But, in this equation, V and W are matrices with two columns with the restriction that the second column is the complex conjugate of the first one. The resulting eigenvalue problem is given in (3.16).

$$(A_r + E_r U V^*) U T = (E_r + E_r U W^*) U T D_u \tag{3.16}$$

It is easily shown that these updates do not introduce complex entries in the reduced matrices. As shown in the appendix A, the new eigenvectors are UT and the entries of the diagonal matrix are given by the eigenvalues of the matrix Ω in (3.17). Its 4 entries are explicitly shown for further development.

$$\Omega = T D_u T^{-1} = \frac{1}{\Delta} \begin{bmatrix} \alpha & \beta \\ \beta^\dagger & \alpha^\dagger \end{bmatrix} = (\mathcal{I} + W^* U)^{-1} (D + V^* U) \tag{3.17}$$

The terms α and β are given by (3.18). The value of delta can be obtained by (3.19).

$$\begin{aligned} \alpha &= (1 + p^\dagger)(\lambda + r) - q^\dagger s \\ \beta &= (1 + p^\dagger) s^\dagger - q^\dagger (\lambda^\dagger + r^\dagger) \end{aligned} \tag{3.18}$$

$$\Delta = \frac{1}{|1 + p|^2 - |q|^2} \tag{3.19}$$

The four constants used in these definitions are given by (3.20). In these equations, u, v and w are the first columns of their respective matrices.

$$\begin{aligned} r &= v^* u \quad p = w^* u \\ s &= v^T u \quad q = w^T u \end{aligned} \tag{3.20}$$

All of these recursive definitions will prove to be very useful in the final formulas for the stabilization method. As have been proved in the appendix A, the eigenvalues of Ω are given by (3.21).

$$\lambda_u = \frac{1}{\Delta}\left(\mathrm{Re}(\alpha) \pm \sqrt{\mathrm{Re}(\alpha)^2 - |\alpha|^2 + |\beta|^2}\right) \tag{3.21}$$

In the same manner that has been done to the rank-1 update method, it is necessary to identify how a rank-2 update of the subspace produces an update of the reduced matrices. This can be done by expanding the updated eigenvalue problem in (3.22). The matrices L and Q represent a rank-2 update on $\mathcal{W}_0$.

$$(W_0 + LQ^*)^* A\mathcal{V}UT = (W_0 + LQ^*)^* E\mathcal{V}UTD_u \tag{3.22}$$

Doing the multiplications and gathering the reduced matrices as one term results in (3.23).

$$(A_r + \underbrace{QL^*A\mathcal{V}}_{UV^*})UT = (E_r + \underbrace{QL^*E\mathcal{V}}_{UW^*})UTD_u \tag{3.23}$$

Comparing this equation with (3.16) allows the identification of the matrices V and W as shown in (3.24). For consistence, the matrix Q has to be equal to E_rU.

$$V^* = L^*A\mathcal{V} \qquad\qquad W^* = L^*E\mathcal{V} \tag{3.24}$$

Using these identified values, the constants r, s, p and q can be written as in (3.25).

$$\begin{aligned} r &= l^*A\mathcal{V}u \quad s = l^TA\mathcal{V}u \\ p &= l^*E\mathcal{V}u \quad q = l^TE\mathcal{V}u \end{aligned} \tag{3.25}$$

As the vector l is an arbitrary vector, it can be chosen so that p and q are equal to zero. This simplifies greatly the constants α and β that can be written as in (3.26). As another positive consequence, Δ will be equal to one.

$$\begin{aligned} \alpha &= \lambda + r \\ \beta &= s^\dagger \end{aligned} \tag{3.26}$$

Doing the same as before and using ηl instead of l will not lose generality and allow the writing of the real part of the new eigenvalue equation in function of a scalar quantity. If one choses, as done previously, the new eigenvalues to have real parts equal the negative of the original ones, the value of η must be chosen as in (3.27).

$$\eta = -\frac{2\,\mathrm{Re}(\alpha)}{\mathrm{Re}(l^* A \mathcal{V} u)} \tag{3.27}$$

The Algorithm 5 summarizes this section and show how to compute the update to fix an unstable complex conjugate eigenvalue.

Algorithm 5 Update to remove two complex conjugate unstable eigenvalues

procedure $\mathcal{W}$ = Remove Unstable Eigenvalue($\mathcal{V}$, $\mathcal{W}$, A, E, E_r, u, λ)

$q \longleftarrow E_r u$
$l \longleftarrow$ random vector
$p \longleftarrow$ orthogonal vector$\left(\left[\mathcal{W} \;\; E\mathcal{V}u \;\; (E\mathcal{V}u)^{\dagger}\right], l\right)$
$r \longleftarrow l^* A \mathcal{V} u$

$\alpha \longleftarrow \lambda + r$
$\eta \longleftarrow -2 * \frac{\Re(\alpha)}{\Re(l^* A \mathcal{V} u)}$

$l \longleftarrow \eta l$
$\mathcal{W} \longleftarrow \mathcal{W} + 2 * \Re(l q^*)$

return $\mathcal{W}$
end procedure

Having these two processes in hand, one can construct a bigger algorithm to produce stable Reduced Order Models of any nature. This is shown in the Algorithm 6. The result obtained by applying this process is going to be briefly demonstrated at the end of this chapter and further used in Chapter 4.

Algorithm 6 Producing Stable Reduced Order Models

procedure $\mathcal{W}$ = Stabilize System(A, E, $\mathcal{V}$)
  $\mathcal{W} \leftarrow \mathcal{V}$
  $\Lambda \leftarrow \lambda\,(\mathcal{W}^T A \mathcal{V}, \mathcal{W}^T E \mathcal{V})$
  $\Lambda_R \leftarrow \{\lambda_i \in \Lambda : \Im\{\lambda_i\} = 0\}$
  $\Lambda_I \leftarrow \{\lambda_i \in \Lambda : \Im\{\lambda_i\} \neq 0\}$
  while $|\Lambda_R| \neq 0$ **do**
    $\lambda_r \leftarrow$ Choose(Λ_R)
    $\mathcal{W} \leftarrow$ update($\mathcal{W}$, λ_r)
    $\Lambda_R \leftarrow \Lambda_R \backslash \{\lambda_r\}$
  end while
  while $|\Lambda_I| \neq 0$ **do**
    $\lambda_i \leftarrow$ Choose(Λ_I)
    $\mathcal{W} \leftarrow$ update($\mathcal{W}$, λ_i)
    $\Lambda_I \leftarrow \Lambda_I \backslash \{\lambda_i, \lambda_i^\dagger\}$
  end while
  return $\mathcal{W}$
end procedure

3.5 Adaptive Placement of Expansion Points

The moment matching technique that was presented in the last sections has the expansion points as an input parameter. As each expansion point grantees only a local accuracy, bad choices of these values may produce poorly Reduced Order Models for the whole frequency range. If one knows the system that is being analyzed, it can use this knowledge to approximate the placement of the points of expansion that will produce accurate models. However, this is not the case in many application and a general method is necessary. We have developed an adaptive way to choose where to place the expansion points based only in the knowledge of a frequency range of interest.

3.5.1 Adaptive Algorithm

As previously seen, the reduced order model obtained by moment matching will have the same moments as those of the High Fidelity Model. The 0-th moment for

a given expansion point is the frequency response for the system in the frequency s_0. This can be directly verified by plugging 0 in the value of k in (2.47). This means that the frequency response of both reduced and high fidelity models are the same at that point. This is the main idea behind the following heuristic, shown as Algorithm 7 and detailed explained in the following paragraphs.

Algorithm 7 Automatic Placement of Expansion points

```
1:  procedure red = Placement of Expansion points(sys, m0, m, fl, fu, α, δ)

2:      Stack.push([f_l, f_u])
3:      red ← reduce(SYS, [f_l, f_u], m_0)

4:      while Stack not empty do
5:          interval ← Stack.pop()
6:          cutPoint ← interval.lower ^(1 - α) * intervalo.upper ^α
7:          redTry ← reduce(SYS, [red.freqs, cutPoint], m)
8:          int1 ← [interval.lower, cutPoint]
9:          int2 ← [cutPoint, interval.upper]
10:         if max( | simulation(redTry) - simulation(red) | > δ) ∈ int1 then
11:             Stack.push(int1)
12:             Keep ← true
13:         end if
14:
15:         if max( | simulation(redTry) - simulation(red) | > δ) ∈ int2 then
16:             Stack.push(int2)
17:             Keep ← true
18:         end if
19:
20:         if Keep is true then
21:             red ← redTry
22:         end if
23:     end while

24:     return red
25: end procedure
```

The user must supply lower and upper bounds for the frequency range of interest and an acceptable error tolerance. These bounds are used to build the first frequency interval. Each extreme receives one expansion point with m_0 moments matched. The default value is $m_0 = 2$. This will produce a first reduced order model of very low order but that is very good around the extremes of the supplied interval. This model and its frequency response are placed in a stack (FIFO data structure) and an iterative process is repeated until a given accuracy is reached. At each iteration, an interval is

received from the stack. A new expansion point is chosen using (3.28), where f_e, f_l and f_u are, respectively, the frequencies of the expansion points, lower and upper values of the interval.

$$f_e = f_l^{1-\alpha} f_u^{\alpha} \tag{3.28}$$

This equation reduces to a geometric mean if α is one half, the default choice. A new reduced order model and its frequency response are obtained after the addition of this expansion point with m moments matched, $m = 1$ being the default. This procedure has delimited two new frequency interval, one between the lower bound and the expansion point and a second one from the expansion point to the upper bound. The difference of the frequency response is computed for these intervals between two different iterations. If any of the values is bigger than the accepted precision, this interval is added to the stack. Otherwise, it is discarded and one considers that the algorithm has converged for that region. It is Important to note that if the interval is discarded, the expansion point is not kept in the final Reduced Order Model.

At this point, the clarification of two processes are very important. The first one is that it is easy to update a reduced order model obtained by projection when the projection matrix $\mathcal{V}$ receives new columns. Equation (3.29) shows the example of the reduction of the matrix A. The biggest part of the final matrix is the same as before. Therefore, this new computation can utilize the old one to be done very quickly.

$$[\mathcal{V}v]^T A [\mathcal{V}v] = \begin{bmatrix} \mathcal{V}^T A \mathcal{V} & \mathcal{V}^T A v \\ v^T A \mathcal{V} & v^T A v \end{bmatrix} \tag{3.29}$$

The second point that must be clarified is how to compute the simulation of the Reduced Order Model in a really quick manner. The simulation of reduced order models will ideally be very fast due to the small size of the resulting matrices. However, one can further improve the simulation speed if a smart factorization of the transfer function is done. This is the subject of the next section.

Very Fast Frequency Domain Simulation

To keep this presentation as simple as possible, one starts with a generic state space representation as in (3.30).

$$\begin{aligned} E\dot{x} &= Ax + Bu \\ y &= Cx \end{aligned} \tag{3.30}$$

The transfer matrix is given by (3.31). The simulation of this system can be done in a naive way by solving the linear system for each value of frequency.

$$H(f) = C(j2\pi fE - A)^{-1}B \tag{3.31}$$

This process will lead to the right answer, but it can be time consuming if the number of frequencies is not small. In the same way that one can use a LU decomposition to solve many linear systems with different right hand sides in a very fast way, it is possible to wonder if the same logic can be applied to the simulation of a dynamic system. The following method is inspired in (Laub, 1980).

In order to do this, a Hessenber decomposition for the matrix pair A and E is done, resulting in the relations in (3.32). In these equations, Q and Z are unitary matrices, $\tilde{A}$ is an upper Hessenber Matrix and $\tilde{E}$ is an upper triangular matrix.

$$\begin{cases} \tilde{A} = QAZ \\ \tilde{E} = QEZ \end{cases} \tag{3.32}$$

Using this decomposition in (3.31), one may obtain an equivalent form as written in (3.33).

$$H(f) = CZ(j2\pi f\widetilde{E} - \widetilde{A})^{-1}QB \tag{3.33}$$

This way of writing the equation is advantageous because the linear system that one must solves involves an upper Hessenberg Matrix. Figure 3.2 illustrates how an addition of an upper Hessenberg matrix and upper triangular matrix results in a Hessenberg matrix.

$$\begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ & * & * & * & * \\ & & * & * & * \\ & & & * & * \end{bmatrix} + \begin{bmatrix} * & * & * & * & * \\ & * & * & * & * \\ & & * & * & * \\ & & & * & * \\ & & & & * \end{bmatrix} = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ & * & * & * & * \\ & & * & * & * \\ & & & * & * \end{bmatrix}$$

Figure 3.2: Scheme showing the nonzero elements of the matrix addition between an upper Hessenberg matrix and an upper triangular matrix

To write (3.33) in a more convenient way, two new matrices are defined as in (3.34).

$$\begin{cases} \widetilde{B} = QB \\ \widetilde{C} = CZ \end{cases} \tag{3.34}$$

Using this new definition, the system transfer function can be written as in (3.35).

$$H(f) = \widetilde{C}(j2\pi f\widetilde{E} - \widetilde{A})^{-1}\widetilde{B} \tag{3.35}$$

This new transfer function has the same structure of the original one (3.31), but has the advantage of involving only upper Hessenberg linear systems. To illustrate how this results in a computational time gain, a model of a cantilever beam of order 480 was simulated for a different number of frequencies. The result is shown in Figure 3.3. It

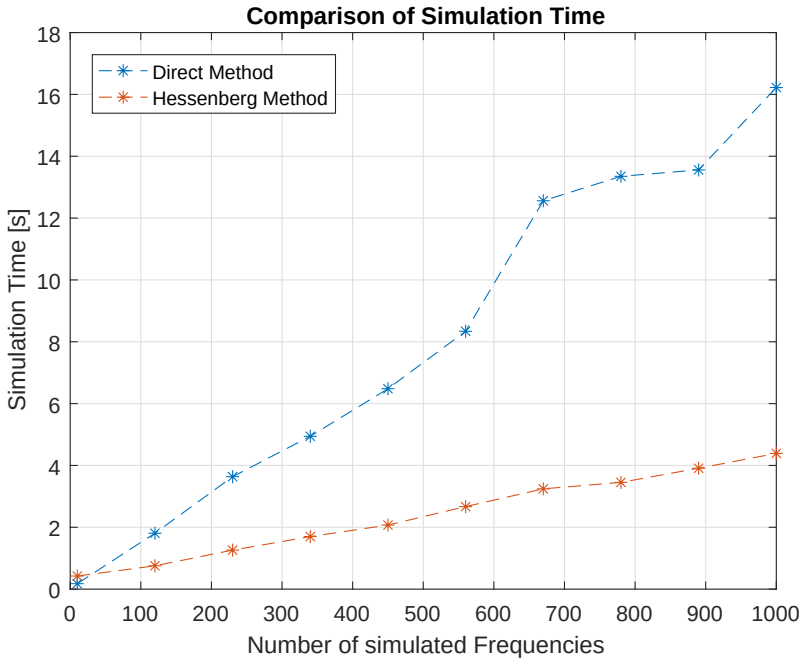


Figure 3.3: Simulation time obtained using the two different methods

is important to say that the time for the matrix decomposition is not negligible and is included in the following graph.

It is important to notice that for the first point of this graphic, corresponding to a simulation of only 10 frequencies, the naive method is faster. This can be understood by the cost of the factorization of the matrices. If one needs only the system response at very few frequencies, this method is more adequate. However, most of the time, a system is simulated for a large number of frequencies. In this case, the second method is much faster than the naive one.

3.5.2 Greedy Algorithm

In the previous process, the placement of the expansion points is only made at the center of intervals that had not yet converged. However, one can think of a strategy to place these points at places were the projection error is maximal. This, however must be done without simulating the High-Fidelity Model.

One way of doing this is by looking at the residue of the approximated state vector of the system. Consider a dynamic system with an impulse function as input, as given in (3.36).

$$E\dot{x} = Ax + B\delta(t) \tag{3.36}$$

The Laplace transform of this set of equations will produce the linear system shown in (3.37).

$$(Es - A)x = B \tag{3.37}$$

Using (2.10), the solution of this system can be approximated, thus allowing the writing of a residue vector for each value of s, as shown in (3.38).

$$r = (E\mathcal{V}s - A\mathcal{V})x_r - B \tag{3.38}$$

The values of x_r can be cheaply obtained for a big number of frequencies due to the low dimensionality of the reduced order model and applying the technique developed in the last section. Hence, at this point, one can cheaply compute the residue vector for each analyzed frequency. The expansion point can then be placed at the frequency that has the residue vector with the biggest norm. The process will continue until the biggest norm of the residue vectors is below a given tolerance. Another possible stopping critter is the verification if the maximal change of the transfer function of subsequent reduced order models is smaller than a given tolerance.

Using the explicit expression for x_r one can show that where the expansion points are placed the residue vector becomes zero. This is a very desirable property because it assures that the process will not loop indefinitely always choosing the same expansion point. To proceed with the demonstration, one assumes that a given expansion point s_0 has been chosen in a previous iteration. Therefore, there exists a vector q that satisfies (3.39).

$$B = (A - s_0E)\mathcal{V}q \tag{3.39}$$

Using this expression, it is possible to further develop (3.38). This is shown in (3.40).

$$\begin{aligned} r &= (s_0E - A)\mathcal{V}(s_0E_r - A_r)^{-1}B_r - B \\ r &= (s_0E - A)\mathcal{V}(s_0E_r - A_r)^{-1}V^T(A - s_0E)\mathcal{V}q - B \\ r &= (A - s_0E)\mathcal{V}q - B \\ r &= (A - s_0E)(A - s_0E)^{-1}B - B \\ r &= B - B = 0 \end{aligned} \tag{3.40}$$

At this point, we have developed methods to produce real, stable and accurate Reduced Order Models. The next section will focus on showing how this can be used to produce a wider variety of Reduced Order Models.

3.6 Example of Application

To illustrate these process, we are going to use a loop antenna obtained by the PEEC method (Freschi et al., 2006).

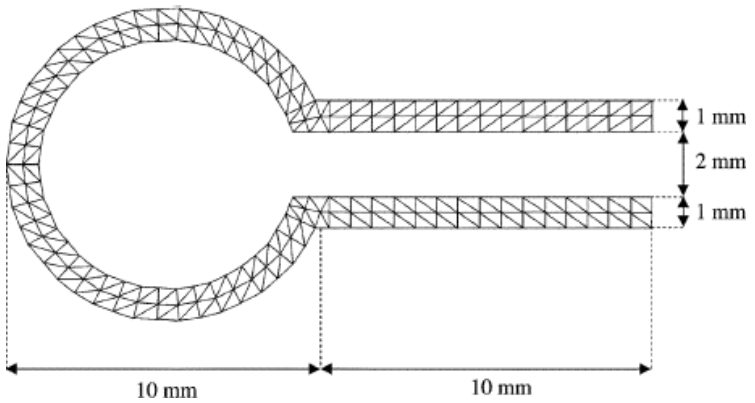


Figure 3.4: Geometry for the loop antenna

Its dimensions can be seen in Figure 3.4. A current is applied as an input signal to its terminals and the voltage difference between these points is measured as output. Therefore, effectively the impedance of the system is being measured. An example of current density distribution on the surface of this system is shown in Figure 3.5.

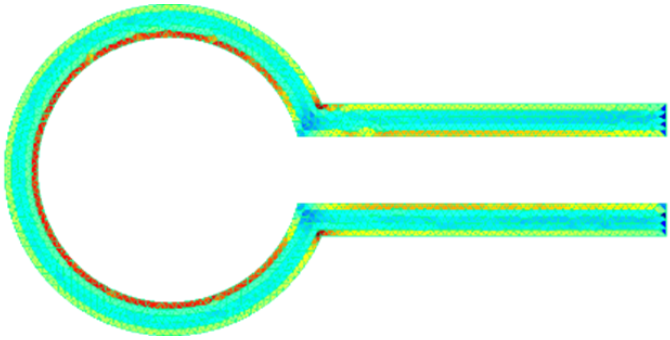


Figure 3.5: Current distribution on the surface of the antenna

The grid has been chosen in a way that the produced system has order 4650. We proceed by showing the result obtained by applying the OFMM with the adaptive technique to match moments and the developed stabilization schema to produce a stable Reduced Order Model. It is very important to have in mind that for the specific case of electrical circuits, methods like PRIMA (Odabasioglu et al., 1997) and SPRIM (Freund, 2011a) can be used to guarantee stability. However, the proposed method is general and works for any system written as a state space in descriptor form.

The reduction process took 39 s and the resulting model has order 18. The simulation time is negligible. The original model took 975 s to be simulated. Therefore, we can conclude that Model Order Reduction has been very efficient in reducing the simulation time. Figure 3.6 shown the end result of this procedure. We will present in more detail the process of expansion point placement in the end of this section.

Figure 3.7 shows the placement of the poles for the original Reduced Order Model. It is clear how a pair has positive real parts. Therefore, the original Reduced Order Model is not stable and may not be simulated in the time domain. This pair of unstable poles were changed by the method that we have developed and have successfully been allocated to the left half side of the complex plane. Another very important point to notice is that the initially stable poles were not changed. This is very important to preserve as much as possible the original system and do not deteriorate the accuracy of the Reduced Order Model during its stabilization.

Using this stabilized model, we can perform time domain simulation of this system. Figure 3.8 presents the result of this process. The input was chosen as a current impulse.

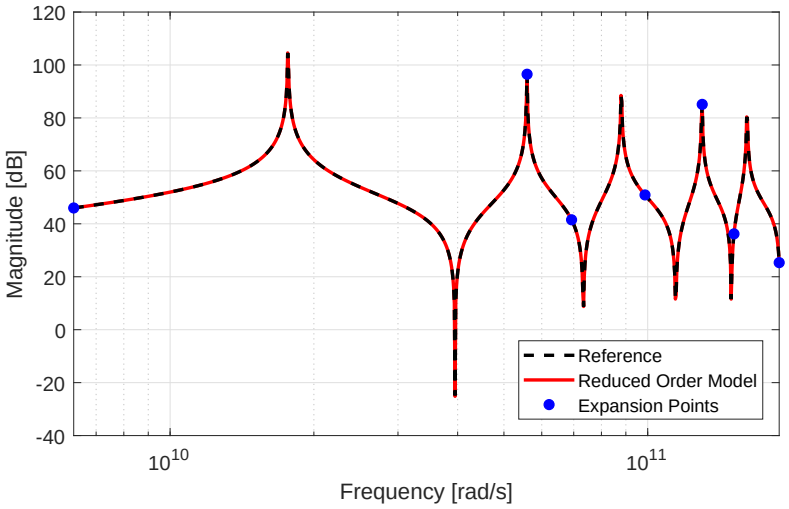


Figure 3.6: Frequency response of the Reduced Order Model.

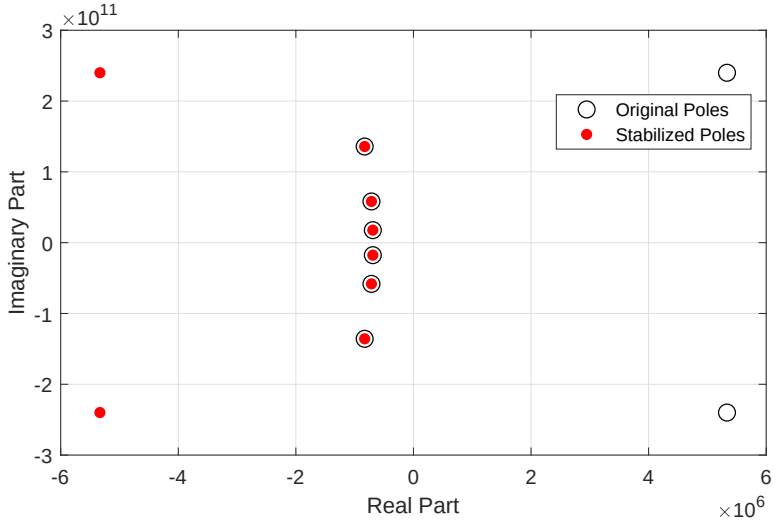


Figure 3.7: Poles for the original and stabilized Reduced Order Models.

Note that the simulation of the High-Fidelity Model is not present. Although the model represents a circuit, some numerical errors can produce unstable systems (Bond and Daniel, 2008a). This is the case for the antenna.

Before ending this section, we present a more detailed view on the automatic placement of expansion points. Figure 3.9 presents this process step-by-step.

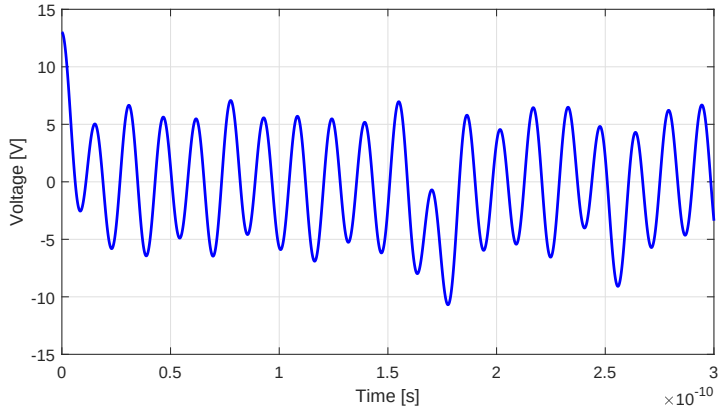
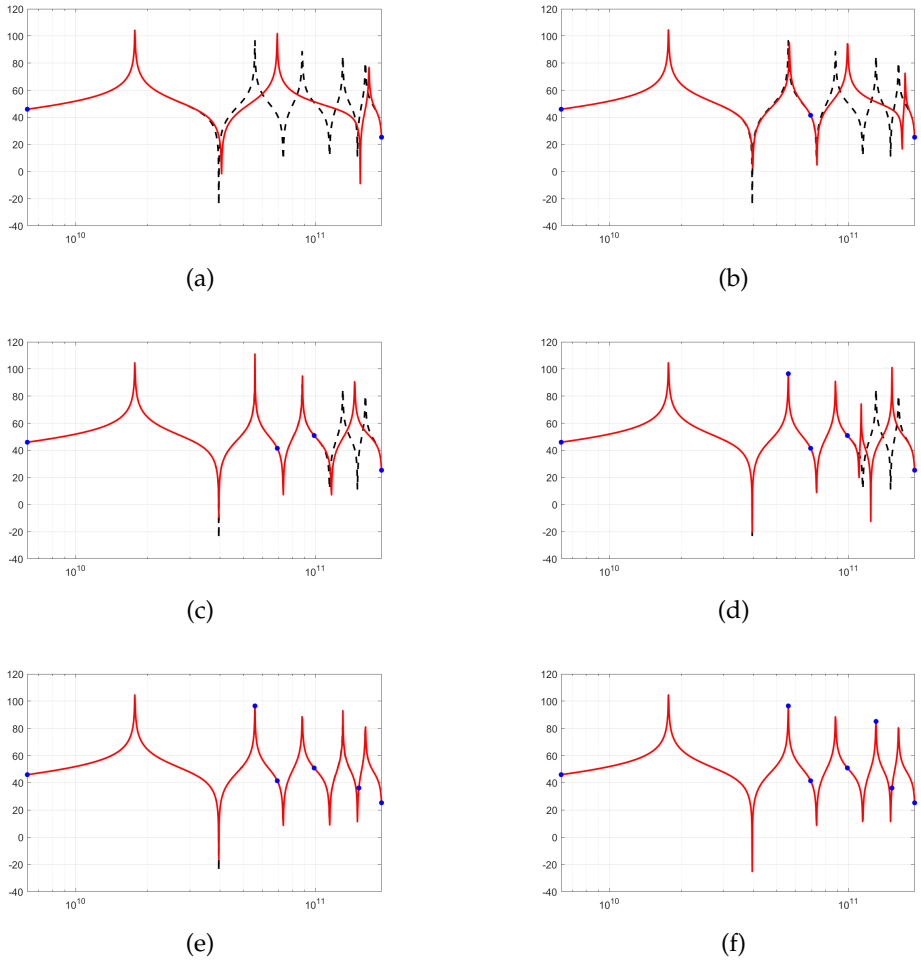


Figure 3.8: Time domain simulation of the stabilized system.



(a) (b) (c) (d) (e) (f)

Figure 3.9: Expansion point placing for the loop antenna. (a) Placement of the points at the limits of the interval. (b)-(e) Placement of the points by the adaptive method. (f) Final position of all the expansion points.

3.7 Conclusion

This chapter presented some improvements over the classic Moment Matching algorithm.

The classical method for computing the Krylov Subspace needed for this method was explained in details and the modifications needed to perform model order reduction were discussed.

The presented contributions allow the computation of real and stable reduced order models in a numerically robust way. An adaptive way of choosing the expansion points was introduced giving, then, to the reader all the needed tools to perform the whole reduction process.

Chapter 4

Exploiting Reduced Order Models for Simulation

4.1 Introduction

In the previous chapters, a cantilever bean and a loop antenna have been used as examples of applications for the diverse Model Order Reduction methods. The cantilever beam has been chosen because it is widely used as a first example in the Model Order Reduction community. It is also easily parametrized serving as a canonical example for methods bases on subspace interpolation. Its use is also very instructive because one sees that the algorithms are general and work well in many different domains of applications.

In this chapter, however, we will introduce two examples of complex electromagnetic models. These models arise from practical applications and will be used to show the application of the Reduced Order Model technique developed in this thesis.

The first one consists on a laminar power bus bar originally presented in (Kuwabara et al., 2016). The model is constructed using the PEEC method and is integrated with MIPSE (acronym for "Modeling of Interconnected Power Systems"), a research project for multi-level and multi-method simulation platform developed in G2ELab. The platform does not produce the system matrices themselves due to the extremely large memory they would require. Only an interface providing products of the matrices times vectors is provided. Therefore, this model allows us to show how one can work with Model Order Reduction taking into account this restriction. One could argue that this is a kind of gray box system.

The second model consists in the two-dimensional modeling of a mammography. It is basically a wave-scattering problem with perfectly absorbing contour conditions.

The modeling is done using Finite Differences in Time Domain. Evidently, the real problem should be modeled in three dimensions. However, this simplified model allows the demonstration of all the main ideas behind Model Order Reduction without an unnecessary extra layer of complexity.

4.2 Laminated Bus Bar Model

The first model that will be simulated using Model Order Reduction is a Laminated Bus Bar. This system is detailedly described in (Kuwabara et al., 2016). It consists of two isolated cooper plates, separated by 4.8mm, with four capacitors placed at given positions. The positive pin of the capacitor is placed in the upper plate and the negative one is pierced through the material and reaches the lower plate. Also connected between the positive and negative plates are the power input and load. Figure 4.1 contains a drawing representing the physical system.

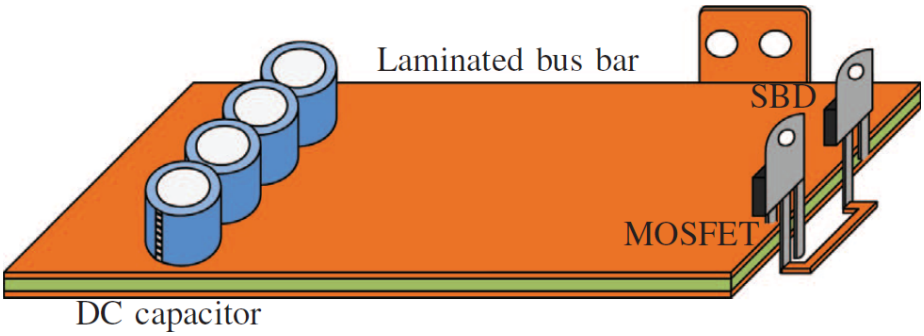


Figure 4.1: Drawing of the laminated bus bar. Adapted from the original (Kuwabara et al., 2016).

To obtain the High Fidelity Model of this system, we have used MIPSE with a PEEC method, an integral formulation of Maxwell equations, to construct the model. This method consists, basically, in the discretization of the space and modeling of each region by an equivalent circuit (Ruehli et al., 2017). This will produces dense matrices for the inductance, capacitance and resistance of the system. In an intuitive level, one may conclude that this is the case by noting that if every single cell of the discretization has a mutual inductance with all the other ones, then a dense symmetric matrix is needed to represent all these interactions.

Dense matrices have large memory requirements and can pose problems even for modern computers. One way of solving this problem is by the use of matrix compression techniques. This is a widely used approach when dealing with PEEC (Li et al.,

2012) (Siau, 2016). In the specific case of MIPSE, the matrices are never explicitly built and the user only has access to the product of the circuit matrices by arbitrary vectors.

Note that in principle one could recover the matrix by performing N multiplications, where N is the size of the matrix. However, this would be completely against the idea of saving memory and in many cases even impossible due to limited resources. Therefore, this restriction influences the way that we build the Model Order Reduction algorithm and is one of the main justifications for the use of Arnoldi based methods. The solve the linear systems necessary to perform moment matching one can use an iterative method. In our specific case we have chosen GMRES (Saad, 2003).

In order to model this system in a simple way, a voltage source has been used as power source and the load is represented by a series association of one inductor and one resistor. Figure 4.2 contains a rought illustration of the model. These choices are not arbitrary. MIPSE, when working with a surface formulation, assumes that all inductors have a series resistance. Therefore, we follow this convention.

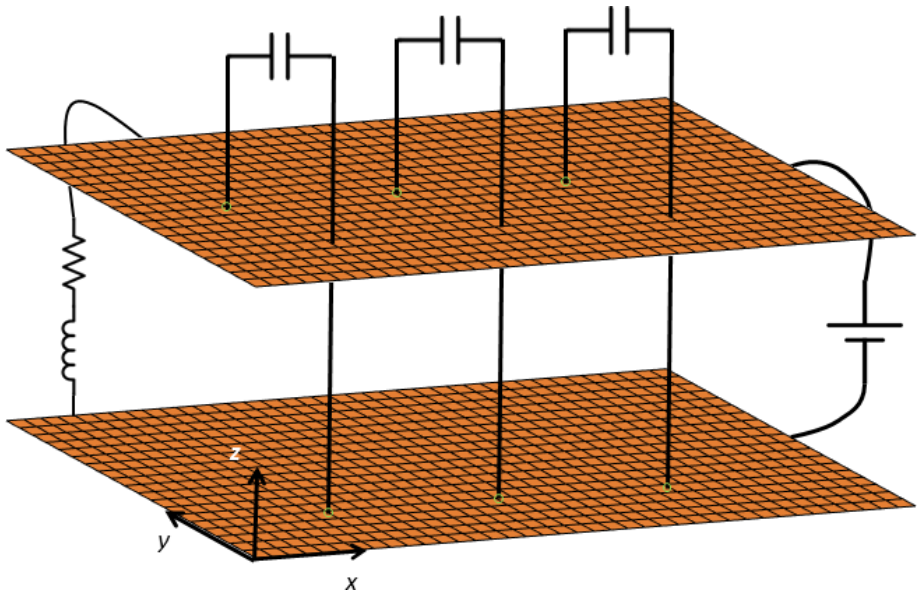


Figure 4.2: Illustration of the power bus bar model.

Having chosen a Cartesian coordinate system and an origin as shown in Figure 4.2, allows the description of the geometry of the circuit in a precise manner. Table 4.1 contains the geometric characteristics of the plates. As they are made of copper, their resistivity is assumed to be 1.68×10^{-8} Ω m and the relative permittivity of the insulator is approximately 4.7.

Characteristic	Value	Unity
Length	1.15	cm
Width	90	mm
Plate Thickness	100	μm
Insulator Thickness	1.6	mm
Separation	4.8	mm

Table 4.1: Geometric characteristics of the two plates.

The connection points for the current source and for the load model are given by Table 4.2. Note that each component has a connection point in the positive and negative plate.

Element	X [mm]	Y [mm]	Z [mm]
Load Model			
positive	51.5	90	4.8
negative	72	90	0.0
Current Source			
positive	115	74.5	4.8
negative	115	45	0.0

Table 4.2: Positioning of the discrete elements on the bus bar

Finally, one must respect the physical space available for placing the capacitors. Figure 4.3 contains the details of their geometry.

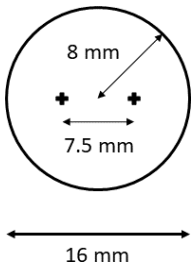


Figure 4.3: Geometry of the capacitor

The Model Order Reduction framework that we have build has been fully codded in Matlab while MIPSE is written in Java. Therefore, an interface layer had to be build allowing the communication of these two platforms. Matlab has a native interface for instantiating Java classes. However, the internal Java version that Matlab is running must coincide with the version used to compile the Java source code. A schematic way

by which the platforms interact can be seen in Figure 4.4.

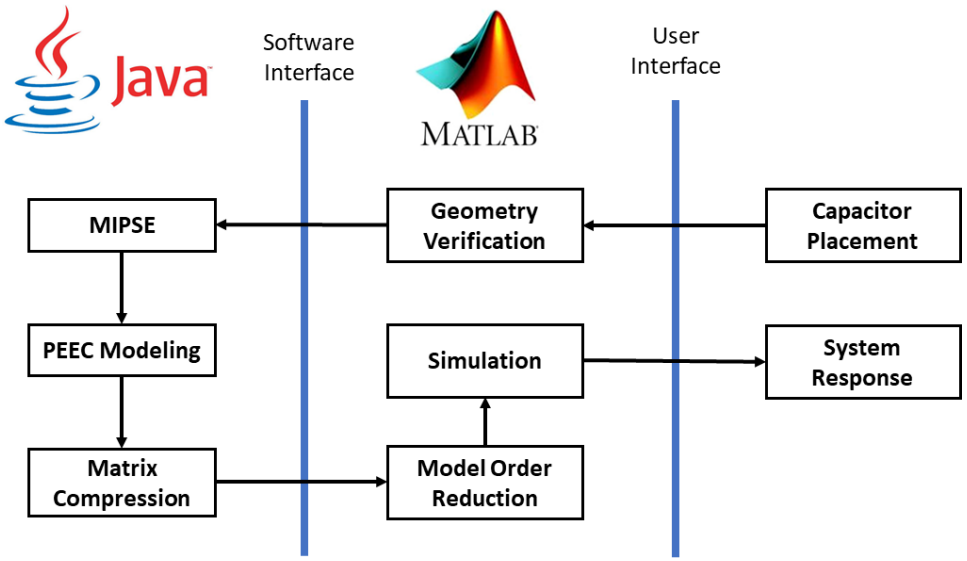


Figure 4.4: Flowchart of the platform integration

The Matlab interface receives a class that is able to perform the products of the following matrices: R, L, C^{-1}, I_{rl}, I_c and I_i. These matrices and the circuit equations are detailedly explained on the Appendix D. However, for the sake of clarity, we briefly review their significance. The matrices R, L and C are the resistance, inductance and capacitance matrices, respectively. Note that we do not have access to the product by matrix C itself, but only to its inverse. The remaining three are incidence matrices. Accordingly to their indexes, they contain information about the connection of segments containing series associations of resistors and inductors, capacitors and current sources.

MIPSE, in its current state, does not automatically integrate the discrete capacitors in C nor the discrete load in R and L. Therefore, this is a task that the intermediate software layer must perform. If N is the number of nodes of the original circuit and n_c is the number of discrete capacitors that we are adding, then we define a matrix $\widetilde{C} \in \mathbb{R}^{(N+n_c)\times(N+n_c)}$ that is called an augmented capacitance matrix. It is obtained by extending the original matrix C by placing the discrete capacitor values at its principal diagonal. This process is shown in (4.1).

$$\widetilde{C} = \left[\begin{array}{c|cccc} C & & & & \\ \hline & c_1 & 0 & 0 & \\ & 0 & c_2 & 0 & \\ & 0 & 0 & c_3 & \\ & & & & \ddots \end{array}\right] \tag{4.1}$$

There is no need to change the matrix I_C, once that the internal circuit builder of MIPSE can already produce the correct incidence matrices taking into account the existence of external discrete elements. The exact same procedure must be performed to incorporate the load model to the system and it can be done in a completely analogous way.

4.2.1 Application

At this point, we have presented everything that is needed to perform the system simulation. We have chosen 4 capacitors each one of 10 mF. Their positions are given in Table 4.3. Accordingly to (Ruehli et al., 2017) the frequency range of interest of this system is from 1 kHz to 10 Mhz. Therefore, we have chosen these values as limits for the adaptive expansion point placement algorithm.

We have chosen the default parameters and selected two moments to be matched at the extremes of the frequency interval and only one moment for the intermediated ones. Figure 4.5 shows the positions of these points. The minimum norm of the residue has been chosen as being 1×10^{-3}.

The original High Fidelity Model has dimensions 4688. This is the result of a desired accuracy of 2mm in each axis, which leads to a discretization grid of 32 by 25. We were not able to go beyond this accuracy due to the limitations that Matlab imposes to the maximum value of memory that its internal java distribution can allocate. It is restricted to 25% of the total memory of the machine.

As we can see in Figure 4.5, the Reduced Order Model was very successful in reproducing the behavior of the High Fidelity One. Note that we have performed the

Element	X [mm]	Y [mm]	Z [mm]
Capacitor 1			
positive	30	9	4.8
negative	30	16.5	0.0
Capacitor 2			
positive	30	30	4.8
negative	30	37.5	0.0
Capacitor 3			
positive	30	51	4.8
negative	30	58.5	0.0
Capacitor 4			
positive	30	72	4.8
negative	30	79.5	0.0

Table 4.3: Positioning of the discrete elements on the bus bar

System	Reduction Time [s]	Simulation Time [s]	Order
High Fidelity Model	–	914	4688
Reduced Order Model	56	< 1	28

Table 4.4: Order and simulation time for the High Fidelity Model and Reduced Order Model.

simulation of the High Fidelity Model only as a base of comparison for the reader. The actual values were never used in our algorithms.

Table 4.4 contains the dimension and time for simulation and reduction. Comparing the values, one can conclude that simulation of the system with the ROM was fast and accurate.

Figure 4.6 contains the time simulation responses for both systems. The input is a square wave of 2 kHz and the output is the voltage measured at the load. One can note that the High-Fidelity Model presents higher frequencies that are not evident in the reduced one. This can be explained when one considers that we have chosen only a small frequency range when performing adaptive moment matching.

We now proceed to the presentation of the results obtained for the second model.

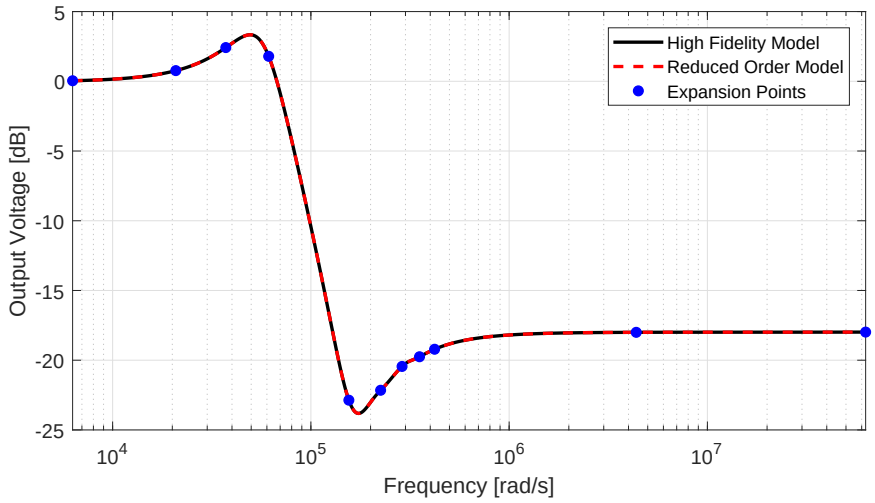
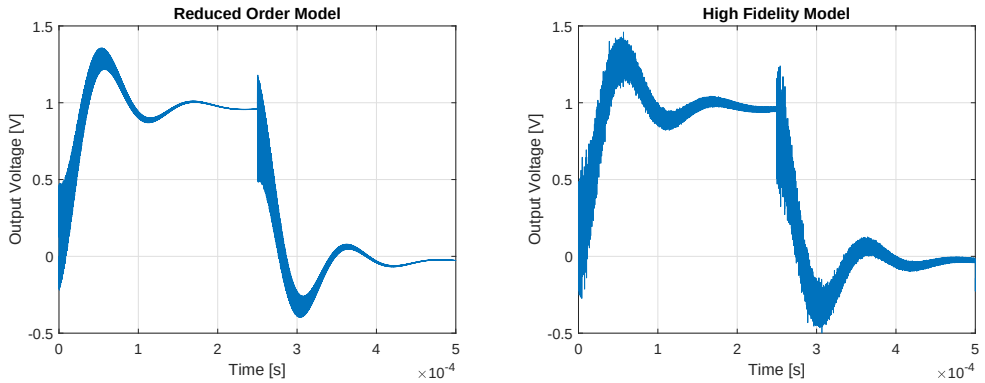


Figure 4.5: Frequency response for the laminated bus bar.



(a) (b)

Figure 4.6: Time response of the system when excited by a 2 kHz square wave (a) Reduced Order Model (b) High Fidelity Model

4.3 Electromagnetic Wave Scattering Problem

In this section, we present the electromagnetic formulation of a wave scattering problem applied to breast imaging (Colgan et al., 2015). It is not, however, the objective of this section to present a full featured introduction to the method of Finite Differences in Time Domain (FDTD) nor to give detailed explanations about Maxwell's equations. The reader can find very good introductory texts on electromagnetic theory at (Feynman et al., 1964) and (Hayt and Buck, 1981). For a very didactic introduction to FDTD we recommend (Schneider, 2010). For a deeper understanding of the subject the reader can refer to (Elsherbeni and Demir, 2016) and (Taflove and Hagness, 2005).

As it is usual in the analysis of any electromagnetic phenomena, one start by writing Maxwell's equations shown in (4.2).

$$\begin{cases} \nabla \cdot B &= 0 & \text{(Gauss Law for Magnetism)} \\ \nabla \cdot D &= \rho_e & \text{(Gauss Law)} \\ \nabla \times H &= j_i + \sigma_e E + \frac{\partial D}{\partial t} & \text{(Ampère's Law)} \\ \nabla \times E &= -\frac{\partial B}{\partial t} & \text{(Faraday's Law)} \end{cases} \tag{4.2}$$

In these equations, the electric and magnetic fields are denoted by E and H. The electric flux density is represented by D and its counterpart the magnetic flux density is denoted by B. For reasons that will become clear in the following paragraphs, we write the volumetric electric charge density and electric conductivity as ρ_e and σ_e. Finally, the impressed current is written as j_i (Balanis, 1999).

Although very recently some researches have been able to find behaviors at a material called "spin ice" that suggest the existence of magnetic monopoles (Fennell et al., 2009), nearly all usages of Maxwell's equations up to this point in time assumes that magnetic monopoles do not exists and, consequently, that there are no magnetic currents. Mathematically, this is represented by the zero in the right hand side of Gauss law for magnetism and by the lack of an impressed current and conductivity terms in Faraday's law.

In doing computer simulations, however, there is no need to attain ourselves to these restrictions. As it will be seen shortly, this will be crucial for obtaining a Perfectly Matching Layer (PML) at the boundary of the domain. In other words, we will simulate a finite region as if it had no boundaries. Including magnetic monopoles and currents, Maxwell's equations can be written in a generalized manner as shown in (4.3).

$$\begin{cases} \nabla \cdot B &= \rho_m \\ \nabla \cdot D &= \rho_e \\ \nabla \times H &= j_i + \sigma_e E + \frac{\partial D}{\partial t} \\ \nabla \times E &= -j_m - \sigma_m H - \frac{\partial B}{\partial t} \end{cases} \tag{4.3}$$

The new quantities with lower indexes m are the magnetic counterpart of the already presented electrical ones. Having these equations at hand, simulation of a specific electromagnetic equipment reduces to solving them respecting the geometry and boundary conditions imposed by the problem. This task can, however, be formidably difficult. This is the reason why many different methods have been developed to solve these equations approximately (Jin, 2015).

One method that is particularly well suited for wave scattering application in rectangular grids is the Finite-Differences in Time-Domain (FDTD) method (Taflove and Hagness, 2005). It consists of using a specific placement of the electric and magnetic fields in the space, known as the Yee cell (Elsherbeni and Demir, 2016), and performing approximations of the time and space derivatives by a central differences schema. This ingenious placement of the electric and magnetic field ensures that the divergence equations are automatically satisfied and only the curl equations have to be solved.

In the two-dimensional case, one possible placement of the fields is shown in Figure 4.7. Note that we have chosen a TM_Z configurations schema. It would also be possible to interchange the electric and magnetic fields forming a TE_Z configuration. The superscripts shall not be regarded as powers, they are simply a notation that will prove to be very useful when writing the state space equations.

In Figure 4.7, there are small empty circles around the analyzed region. These circles are to be interpreted as placeholders for points where the electric field is zero. Therefore, they are not present in the following equations nor in the state matrices. If they were present this would be equivalent to working with a fixed state variable that we know to be zero. Not including these points in the state vector will reflect in the change of some values in the matrix representing the model, as described in (Strang and Aarikka, 1986) and (Strang, 2007), but for a different field of application.

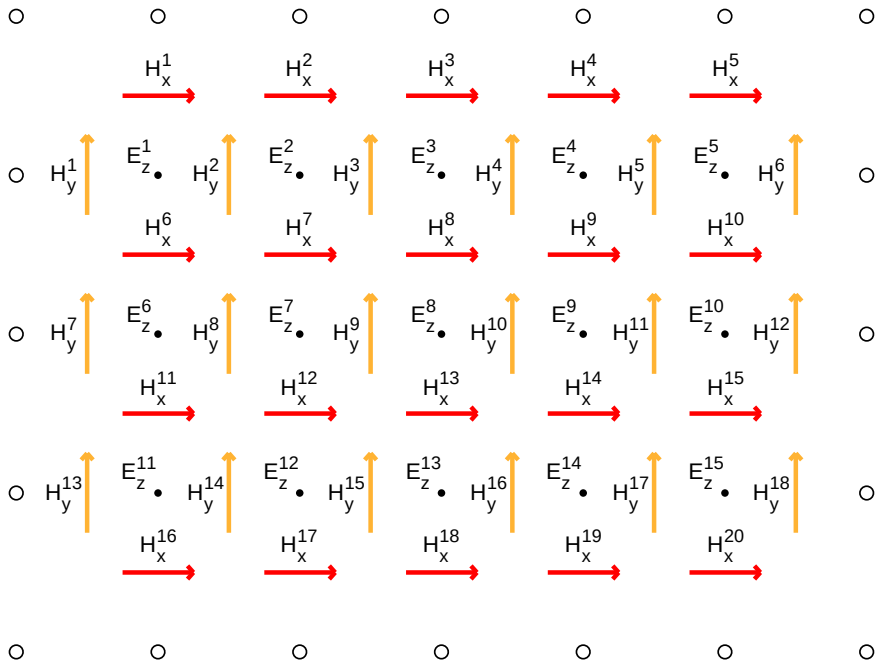


Figure 4.7: Example of field vectors placement using a TM_Z Yee grid.

Writing (4.3) for this specific configuration (TM_Z) and assuming that there are no free charges of any nature and no magnetic current injections leads to (4.4). Note that this does not mean that we went back to the state of (4.2). Magnetic conductivity is still present in (4.4).

$$\begin{cases} \frac{\partial H_y}{\partial x} - \frac{\partial H_x}{\partial y} &= \sigma_e E_z + \epsilon \frac{\partial E_z}{\partial t} + j_{ez} \\ \mu \frac{\partial H_x}{\partial t} + \sigma_m H_x &= -\frac{\partial E_z}{\partial y} \\ \mu \frac{\partial H_y}{\partial t} + \sigma_m H_y &= \frac{\partial E_z}{\partial x} \end{cases} \tag{4.4}$$

To perform wave scattering analysis, we have to simulate a finite domain as if it were infinite. This is normally done by covering the simulation region by a layer that dissipates energy and ideally do not reflect back any waves passing through it. This

method is called the Perfect Matching Layer (PML). There is an approach that is called Convolutional PML that is known to produce better results (Taflove and Hagness, 2005). However, we have not used this method due to its extra complexities when writing the state space.

It can be shown that for the set of equations in (4.4), it is not possible to create a perfectly matching boundary condition (Schneider, 2010). The way that PML solves this problem is by setting $E_z = E_{zx} + E_{zy}$ and solving a different set of equations given in (4.5). The quantities with mixed indexes denote anisotropy in the medium. In regions outside the PML region, $\sigma_{ex} = \sigma_{ey} = \sigma_e$ and $\sigma_{mx} = \sigma_{my} = \sigma_e$.

$$\begin{cases} \epsilon \frac{\partial E_{zx}}{\partial t} + \sigma_{ex} E_{zx} &= \frac{\partial H_y}{\partial x} \\ \epsilon \frac{\partial E_{zy}}{\partial t} + \sigma_{ey} E_{zy} &= -\frac{\partial H_x}{\partial y} \\ \mu \frac{\partial H_x}{\partial t} + \sigma_{mx} H_x &= -\frac{\partial E_{zx}}{\partial y} - \frac{\partial E_{zy}}{\partial y} \\ \mu \frac{\partial H_y}{\partial t} + \sigma_{my} H_y &= \frac{\partial E_{zx}}{\partial x} - \frac{\partial E_{zy}}{\partial x} \end{cases} \tag{4.5}$$

Exactly on the PML layer, the relations (4.6) and (4.7) must be satisfied in order to do not produce reflected waves. When discretizing these equations, however, there will be small reflected waves due to the numerical errors that we have introduced (Elsherbeni and Demir, 2016).

$$\frac{\sigma_{ex}}{\epsilon_0} = \frac{\sigma_{mx}}{\mu_0} \tag{4.6}$$

$$\frac{\sigma_{ey}}{\epsilon_0} = \frac{\sigma_{my}}{\mu_0} \tag{4.7}$$

Equations (4.5) are discretized using central differences in time and space. One important point that must be emphasized is that electric and magnetic field are sampled

at different instants of time, leading to a discretization schema known as "leap-frog". The resulting state space is shown in (4.8). The sub-matrices are constructed to reflect the usual FDTD updating equations, as described in Taflove and Hagness (2005) and Elsherbeni and Demir (2016). As each field only interact with its neighbors, they are very sparse.

$$\begin{bmatrix} \mathcal{I} & 0 & 0 & -E_{ZXHY} \\ 0 & \mathcal{I} & -E_{ZYHX} & 0 \\ 0 & 0 & \mathcal{I} & 0 \\ 0 & 0 & 0 & \mathcal{I} \end{bmatrix} \begin{bmatrix} E_{zx}^{+} \\ E_{zy}^{+} \\ H_{x}^{+} \\ H_{y}^{+} \end{bmatrix} = \begin{bmatrix} E_{ZX} & 0 & 0 & 0 \\ 0 & E_{ZY} & 0 & 0 \\ H_{XEZ} & H_{XEZ} & H_X & 0 \\ H_{YEZ} & H_{YEZ} & 0 & H_Y \end{bmatrix} \begin{bmatrix} E_{zx}^{-} \\ E_{zy}^{-} \\ H_{x}^{-} \\ H_{y}^{-} \end{bmatrix} + \begin{bmatrix} E_{ZJ} \\ E_{ZJ} \\ 0 \\ 0 \end{bmatrix} j \quad (4.8)$$

Note that the state vector in (4.8) is composed of several stacked vectors consisting of the electric and magnetic field sorted by their index, shown previously in Figure 4.7.

The naming convention for the incidence matrices is as follows. The uppercase letter is chosen to coincide with the nature of the field associated with the line: electric (E) or magnetic (H). The first subscripts coincide with the subscripts of the field associated with the line. The last ones coincide with the subscripts of the associated column.

Having these equations in hand, the application of the techniques developed in the previous chapters is straight forward. Note that this system is discrete in time, therefore, the slight modification presented in chapter 2 are necessary to perform Model Order Reduction.

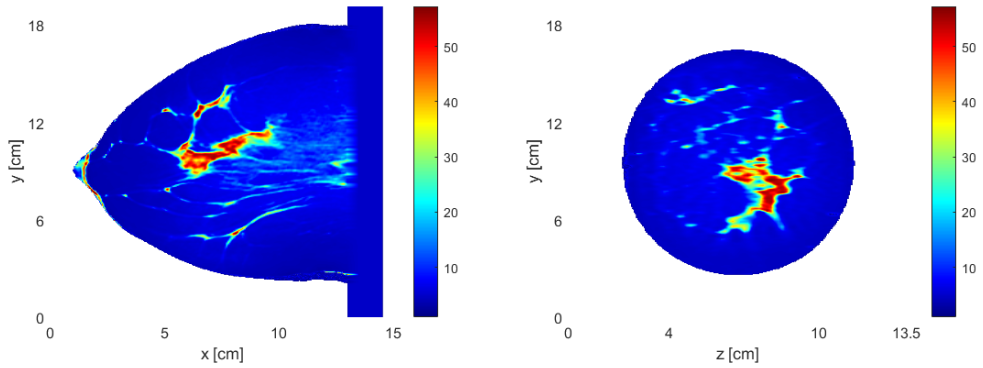
The geometry of the system is specially simple for the 2D problem. It consists of a rectangle with dimensions 14.5 cm by 19.5 cm. The actual breast tissue is contained in an ellipse inside this region. Table 4.5 contains the chosen positions for the emitter and receiver antennas. The origin of the system is chosen at the top left of the domain. Note that in the actual system there are many emitters and receivers. However, for the sake of simplicity, we have chosen only one of each kind.

Antenna	X [cm]	Y[cm]
Emitter	3.0	3.0
Receiver	14.5	3.0

Table 4.5: Position chose for the emitter and receiver.

4.3.1 Application

The data for this system were obtained from the UWCEM Numerical Breast Phantom Repository (Burfeindt et al., 2012). Their format consists of three plain text files containing the information needed to reconstruct the 3D model. Using multidimensional arrays in Matlab, the data can be stored in a very convenient way. To assure that this step has been properly done, we present Figure 4.8. It consists of two cuts of the 3D model. The first one is a side cut known as a sagittal slice. The second one shows the model viewed from above. The colors represent the relative permittivity of the tissue.



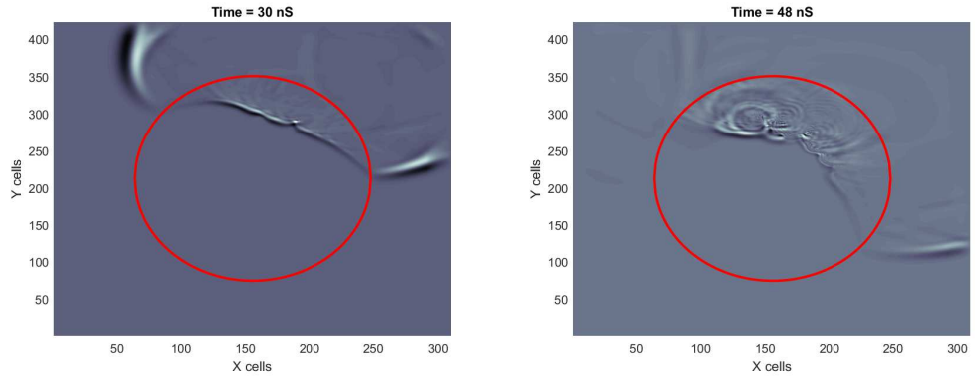
(a) (b)

Figure 4.8: Illustration of the electric properties of breast tissue. (a) sagittal slice (b) top view

We have chosen to use the breast ID 070604PA1. It consists of a grid of 270 by 383 cells when constrained to two dimensions. Therefore, the number of electric field nodes is 131130, the number of magnetic fields in the X direction is 131440 and the number of magnetic fields in the Y direction is 131553. Therefore, the order of the final system is 525253. Note that the matrices are very sparse.

We have performed the time domain simulation of the High Fidelity Model for a small interval of time and the illustration of the result can be seen in Figure 4.9, the red ellipse represents the breast area and the waves on the gray background are representations of the electric field. These images allow us to certify that the behavior of the High Fidelity Model is plausible. Most of the waves go around the breast but a smalls part penetrates and bounces in its interior accordingly to the different electric properties of the material.

We have used the adaptive placement of expansion points to perform moment matching in the frequency domain from 15 MHz to 1.6 GHz. The extremes match 4



(a) (b)

Figure 4.9: Time domain simulation (a) Time = 30 nS (b) Time = 48 nS

moments each and the interior points match 3 moments each. The exact placement of these points can be seen in Figure 4.10.

Finally, we present the comparison of the frequency response in Figure 4.10. Note that differently from other systems, this result presents some visible imperfections. Table 4.6 contains the order of the systems and the time it took for the reduction and simulation.

System	Reduction Time [s]	Simulation Time [s]	Order
High Fidelity Model	–	3003	525253
Reduced Order Model	990	< 1	70

Table 4.6: Order and simulation time for the High Fidelity Model and Reduced Order Model.

Before the end of this section, we emphasize that the simulation and reduction of these complex models were only capable due to the methods developed in this thesis. The models are accurate and stable, allowing the time domain simulation. Even in the case of circuits, where instabilities are theoretically not a problem, practical numerical factors may produce unstable system (Bond and Daniel, 2008b). Therefore, our method is useful even in these cases.

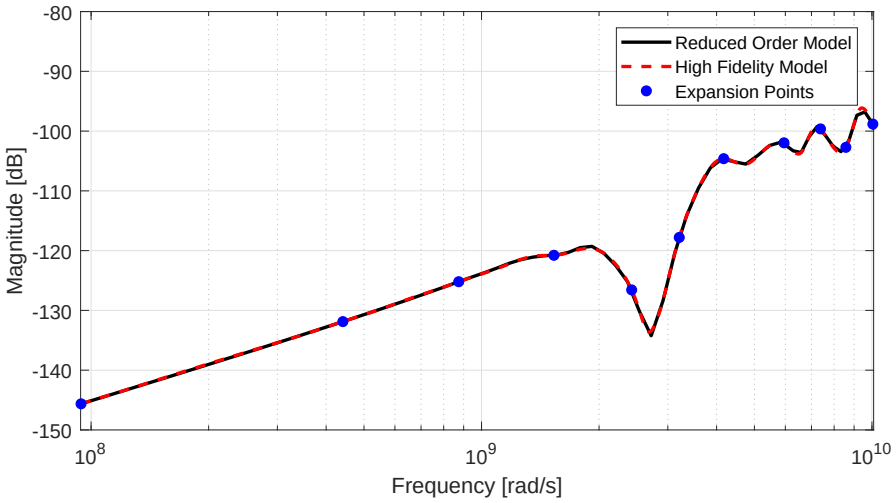


Figure 4.10: Frequency response obtained at the receiver for the wave scattering problem.

4.4 Conclusion

In this chapter, we have introduced two complex electromagnetic models, obtained with different formulations. The first one used the PEEC methodology to model a power bus bar and the second one uses FDTD to model a mammography application. Both systems have direct real world applications and are of very large order.

We have shown how to successfully use Reduced Order Models to obtain accurate time domain simulations of these two systems, validating the methodology and implementation that we developed in the previous chapter.

The next chapter presents some techniques which allow the use of these models in optimizations processes.

Chapter 5

Optimization over Reduced Order Models

5.1 Introduction

This chapter presents some techniques that allows the use of Reduced Order Models in optimization processes. In the same manner that we did in Chapter 2, the current one reviews some known algorithms and explains how they work. This will prove to be important to understand the optimization procedure performed in Chapter 6.

For the sake of completeness we present the well known formulation of an optimization problem. Its statement for only one single objective is presented in (5.1).

$$\begin{cases} \text{minimize} & f(x) \\ \\ \text{subject to :} & g_i(x) \leq 0 \quad i = 1, \ldots, r \\ & h_j(x) = 0 \quad j = 1, \ldots, s \end{cases} \tag{5.1}$$

The vector $x \in \Omega \subset \mathbb{R}^n$ is sometimes called the decision vector, containing the n decision variables. The set Ω is the collection of all the possible decision vectors and is called the search space. The functions $g_i : \Omega \to \mathbb{R}$, for $i = 1, \ldots, r$, and and $h_j : \Omega \to \mathbb{R}$, for $j = 1, \ldots, s$, are the r inequality and s equality constraints, respectively. Finally, $f : \mathcal{R}^n \to \mathbb{R}$ is the objective function that associates $f(x)$ with each decision vector x. A solution is called a global optimum if its objective value is less or equal to the objective

value of any other possible decision vector in Ω and all the constraints are satisfied. A local optimum is a point that obeys the former definition but restricted to an open subset of Ω instead of the whole space.

Among the different methods present in the literature for the solution of the presented optimization problems (Newton's Method, Conjugate Gradient, Genetic Algorithm, Interior Points (Fletcher, 2013), (Goldberg, 1989), (Boyd and Vandenberghe, 2004)), all of them share one unifying characteristic: they have to evaluate the objective function at many different points on the search space. Each evaluation can be very time consuming if a complex computer simulation has to be run. Therefore, it is natural to try to use Model Order Reduction to accelerate this process.

One natural way to proceed would be to simple evaluate the objective function using the Reduced Order Model in the place of the High Fidelity one. However, there are methods that can do even better.

In the next sections two different kind of approaches to the very fast computation of objective functions are presented. The first kind perform the direct interpolation of the objective function that can be obtained by performing any kind of analysis on a Reduced Order Model. The second kind tries to guess the complete Reduced Order Model for a unsampled parameter set. Therefore, this second kind is able to produce models that can be simulated and integrated with external software, while the first kind only return scalar values.

5.2 General Purpose Interpolation Methods

As said in the last section, optimization will sample the search space in many different points while searching for an optimum. Knowing some points of the objective function, it should be possible to give an approximation of the value of the objective function for points that were not sampled.

Using Moment Matching with automatic expansion point placements, it is possible to use the Reduced Order Models output to sample the objective space at some given positions and regard the obtained values as exact or extremely good approximations. Points for different parameter sets can be obtained by interpolation.

There are many highly successful interpolation techniques such as cubic spline and Laplace interpolation (Press, 2007) that depend on points to be arranged in a

structured manner. In the optimization process, however, it is very likely that the sampled points will not lay on a grid. Therefore, one needs an algorithm capable of dealing with scattered data. The next sections present two famous approaches: Radial Basis Interpolation and Kriging.

5.2.1 Radial Basis Interpolation

Radial Basis Interpolation is first presented due to its simplicity. The next section will present Kriging, a much more complex approach.

Knowing the value of a function $y(x) : \mathbb{R}^p \to \mathbb{R}$ at n given points $(x_0, \ldots, x_n)$, this algorithm aims at producing an approximation to an unknown position x_0. This value is denoted $y(x_0)$. The idea is to represent the original function by a weighted sum of simpler ones. This sum is shown in (5.2).

$$y(x) = \sum_{k=1}^{n} \beta_k \phi_k(x) \tag{5.2}$$

Each function $\phi_k(x)$, known as Radial Basis Function (RBF), is normally chosen to be of the form $\phi_k(x) = \psi_k(|c_k - x|)$, where c_k are vectors called the centers of the functions. Therefore, each one can, in principal, be different. However, ψ_k are most of the time chosen to have the same form. Therefore, points having the same distance from c_k will evaluate to the same value. In this application, c_k will be chosen to equals x_k.

Enforcing that the interpolated value should match exactly the ones at the known points results in (5.3).

$$y(x_i) = \sum_{k=1}^{n} \beta_k \phi_k(x_i) \tag{5.3}$$

Defining y and β as being the vector containing, respectively, the values of y_i and β_i and the matrix $\tilde{\phi} = \left[\phi_j(x_i)\right]_{ij}$ allows the writing of (5.3) in a much more convenient way, shown in (5.4).

$$y = \tilde{\phi}\beta \tag{5.4}$$

The weighting coefficients can be obtained by solving the linear system and the approximated value is computed by (5.2).

There are many different radial basis functions that can be shown to produce invertible and sometimes even positive definite $\tilde{\phi}$ matrices (Buhmann, 2003). As an example, we have equations (5.5), (5.6) and (5.7) that are known as thin plate spline, multiquadric and Gaussian, respectively.

$$\phi(d) = d^2 \ln(d) \tag{5.5}$$

$$\phi(d) = \sqrt{1 + \epsilon^2 d^2} \tag{5.6}$$

$$\phi(d) = e^{-\epsilon^2 d^2} \tag{5.7}$$

To exemplify the application of the method, we take advantage of the easy way in that the cantilever beam model can be parametrized. We have changed its length from 1 meter up to 2.5 meters and sampled the displacement obtained for the frequency of 10 Hz. Figure 5.1 shows the original function and the interpolated one. For this process we have used the function (5.7) with $\epsilon = 0.1$.

Radial Basis Interpolation is very simple mathematically and can be used to interpolate the objective function of a very time consuming optimization process. Therefore, it can be used to reduce the computational time. However, it does not present error bounds for the values that have been predicted. In the next section we present Kriging a technique that is more complex but that overcomes this problem.

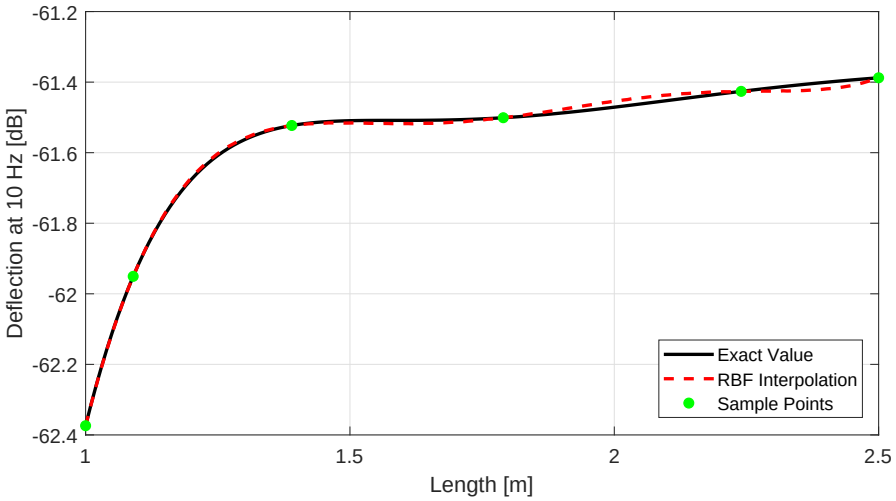


Figure 5.1: Interpolated function obtained by RBF interpolation

5.2.2 Kriging

Kriging is a method whose origins are connected to the geostatistics community (Isaaks et al., 1989). It is normally used to infer values of some soil property or mineral concentration at all points in space given only a few sampled sites. Examples of such applications are the determination of organic matter and coal concentration in the soil (Clark and Harper, 2000).

The version of Kriging that is presented in this section follows the work of (Sacks et al., 1989) and (Jones et al., 1998). In the optimization community, it is known as Design and Analysis of Computer Experiments (DACE). This section aims, however, at presenting the algorithm to the reader in a didactic manner. Therefore, many mathematical details have been omitted. However, they can be found at the Appendix E.

There is a free and very useful Matlab toolbox that implements this algorithm, also called DACE (Lophaven et al., 2002). The results that we present are, however, made with our own computer code. This allows us to perform highly optimized integration with the algorithm presented in Chapter 5. Another reason for this choice is that DACE can sometimes produce negative values for the variance of the predicted error (Lophaven et al., 2002).

Suppose that there is a function $y : \Omega \subset \mathbb{R}^n \rightarrow \mathbb{R}$ that is unknown or which computational cost is extremely high. At a certain point of the optimization algorithm, p points

$(x_i, \ldots, x_p)$ of the design space have been sampled and their p objective function values $(y(x_i), \ldots, y(x_p))$ are known. These results are obtained, normally, by very expensive computer simulations and can be very time consuming.

The main goal is to obtain an estimation of the value $\hat{y}(x_0) \in \mathbb{R}$ at the unsampled point $x_0 \in \mathbb{R}^n$. Assume that $\hat{y}(x_0)$ can be written as a linear combination of the p known values. Naming the constants of this linear combination $\beta_1, \ldots, \beta_p$, one can write this combination as (5.8). In this equation, β and y are column vectors composed of the individual values of β_i and $y(x_i)$.

$$\hat{y}(x_0) = \sum_{i=1}^{p} \beta_i y(x_i) = \beta^T y \tag{5.8}$$

The exact prediction error $\delta \in \mathbb{R}$ is then given by (5.9).

$$\delta = \hat{y}(x_0) - y(x_0) = \beta^T y - y(x_0) \tag{5.9}$$

It is impossible to exactly evaluate this function because it would require knowing the value of $y(x_0)$. To tackle this problem, the Kriging method consists in assuming that the values of $y(x_i)$ are realizations of a random variable Y of the form shown in (5.10) and requiring no bias and minimal variance of the estimation error.

$$Y(x) = \left(\sum_{i=1}^{r} \alpha_i f_i(x) \right) + Z(x) \tag{5.10}$$

In this equation, α_i are called the regression coefficients and the functions f_i are called the regression functions. One should not confuse these functions with the objective function defined in the optimization problem formulation. Henceforth, $\alpha \in \mathbb{R}^r$ is the vector whose entries are α_i and $f(x_j) \in \mathbb{R}^r$ is defined as being the column vector

containing the values $f_i(x_j)$ as its entries. Note that the placement of the index is crucial and that f_i and y are completely different entities. The random process Z has zero mean and a covariance given by (5.11). In this equations, σ is a positive scalar and $R(a,b)$ is the correlation between two sampled points a and b.

$$\text{cov}\{a, b\} = \sigma^2 R(a, b) \tag{5.11}$$

Before continuing, some notations will be defined in order to keep further exposition as simple as possible:

- $F \in \mathbb{R}^{p\times r}$ is defined as being the transpose of the matrix obtained by placing the vectors $f(x_i)$ side by side.
- $R \in \mathbb{R}^{p\times p}$ is defined as $[R]_{ij} = R(x_i,x_j)$.
- $r \in \mathbb{R}^p$ is defined as $r_i = R(x_i,x_0)$.

In the following paragraphs, the mathematical detail have been omitted in order to keep the text as readable as possible. However, all the intermediate details can be found on the Appendix E.

Using the model (5.10), allows the writing of the stochastic model of the error δ as in (5.12). To simplify the notation, Y is a column vector containing the random variables $Y(x_i)$.

$$\Delta = \beta^T Y - \hat{Y}(x_0) \tag{5.12}$$

Imposing that the mean value of the error should be zero for an arbitrary vector α, results in (5.13).

$$F^T\beta = f(x_0) \tag{5.13}$$

This condition assures that the expected value of the error will be zero. The second condition imposed by the Kriging method is that the variance of the error, given by (5.14) should be minimal.

$$\text{var}\{\Delta\} = \sigma^2(1 + \beta^T R\beta - 2\beta^T r) \tag{5.14}$$

To obtain the set of equations that the prediction coefficients must satisfy, one must solve the optimization problem in (5.15) for β. Note that the constant and the multiplicative factor has been taken out of the expression once they do not alter the optimal point.

$$\begin{cases} \text{Minimize :} & \beta^T R\beta - 2\beta^T r \\ \text{Subject to :} & F^T\beta = f(x_0) \end{cases} \tag{5.15}$$

We know that this problem has a minimum due to its quadratic nature and the fact that R is positive definite. It can be solved by applying the method of Lagrange multipliers, denoted by $\lambda \in \mathbb{R}^r$. The linear system that the solution obeys is given by (5.16).

$$\begin{bmatrix} R & F \\ F^T & 0 \end{bmatrix} \begin{bmatrix} \beta \\ \lambda \end{bmatrix} = \begin{bmatrix} r \\ f(x_0) \end{bmatrix} \tag{5.16}$$

The solution of this system of equations will provide the values of β. There is, however, no information in this system that allows the determination of the values for σ (variance of the error), α (coefficients that multiply $f_i(x)$) and R (correlation matrix). These quantities are crucial in the utilization of the prediction algorithm and must be determined.

In order to do so, one can start by computing the maximum likelihood estimation for α, denoted $\hat{\alpha}$, which results in (5.17).

$$\hat{\alpha} = (F^T R^{-1} F)^{-1} F^T R^{-1} y_s \tag{5.17}$$

Using this result, one can obtain the maximum likelihood estimation for σ^2, which is similarly denoted by $\hat{\sigma}^2$ and given by (5.18).

$$\hat{\sigma}^2 = \frac{1}{n}(y - F\hat{\alpha})^T R^{-1}(y - F\hat{\alpha}) \tag{5.18}$$

The values of R are obtained by the introduction of a correlation model. In (Sacks et al., 1989) the authors use the function $\mathcal{K}$ shown in (5.19). Note that it is symmetric in its arguments as our intuition predicts.

$$\mathcal{K}(a, b) = \sum_{i=1}^{n} \theta_i |a_i - b_i|^{\psi_i} \tag{5.19}$$

The matrix R can now be obtained by applying this function to the pairs of sampled points. There is, however, one thing that prevents this in a first analysis. Some new parameters $\theta_i \in \mathbb{R}_+$ and $\psi_i \in [1, 2]$ have been introduced. Their values depend of the data and shall be obtained by the use of a last maximum likelihood estimation process.

The concentrated likelihood function for the sampled data is given by (5.20). Note that the dependence of θ and ψ is through $\hat{\sigma}$ and R. We recall that p is the number of sampled points.

$$\mathcal{L}(\theta, \psi) = \frac{1}{\det(R)(2\pi\hat{\sigma}^2)^p} \exp\left(-\frac{1}{2\hat{\sigma}^2}(y - F\hat{\alpha})^T R^{-1}(y - F\hat{\alpha})\right) \tag{5.20}$$

Optimizing this complicated function is surprisingly equivalent to optimizing 5.21, where L is the Cholesky factor of the matrix R. One can solve this problem by applying genetic algorithms or by restarted Newton-Like methods.

$$\widetilde{\mathcal{L}}(\theta, \psi) = \hat{\sigma}^2 \det(L)^{\frac{2}{p}} \tag{5.21}$$

Having found all the unknown quantities, we can compute β by (5.16) and use (5.8) to approximate the values for unsampled points. We present a simple example of usage in Figure 5.2. It consists in interpolating the same function used for the RBF case. It consists in the deflection obtained for the cantilever beam when setting different lengths.

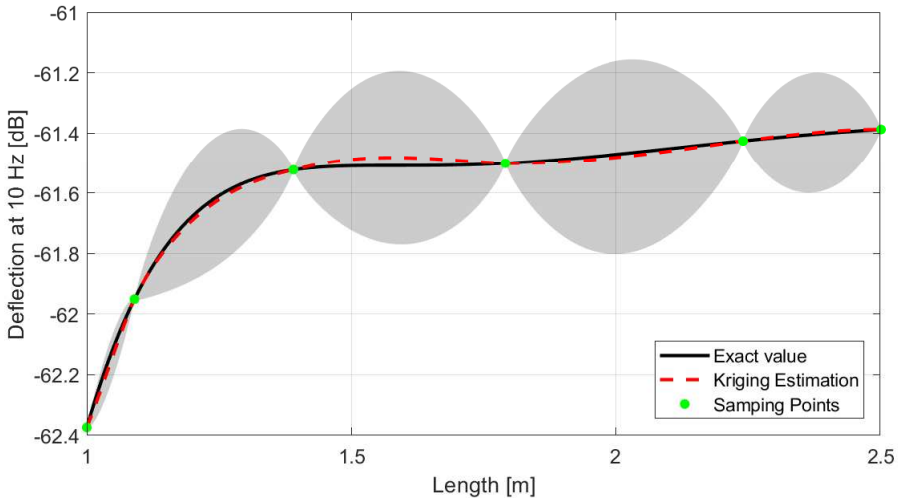


Figure 5.2: Example of the Kriging estimation

The shaded region represents the 95% confidence interval for the interpolation. This is one of the main advantages of Kriging over RBF interpolation: it allows us an easy way to measure how good is our result.

One problem that has not been addressed in this section is the choice of the sampling points. For the current example it has been done arbitrarily. However, it will be shown in Chapter 6 how to perform adaptive sampling of the search space.

Before presenting two different methods, we emphasize that RBF interpolation and Kriging can be used to replace very expensive objective functions in optimization, allowing faster computations. The main disadvantage of those methods when compared to the following ones is that they only work with scalar values as black-boxes. However, in this thesis, the objective functions are being calculated by using Reduced Order Models. Therefore, this additional information can be used to produce useful results. The following methods will show ways to use this in order to produce not interpolated values, but Interpolated Reduced Order Models.

5.3 Interpolation of Reduced Order Models

In this section, we present some techniques that are conceptually different from Kriging and Radial Basis Interpolation. Matrix Interpolation and Subspace Interpolation will not only produce the value of the objective function at an unknown positions, they will construct the Reduced Order Models that represent the High Fidelity Model at unsampled parameter values.

This will be very convenient not only to perform system simulation in the context of optimization but also to evaluate any constraints that have to do with a specific dynamic behavior or acceptable amplitudes in the frequency domain.

5.3.1 Subspace Interpolation

Parametric Model Order Reduction by subspace interpolation has been introduced by (Amsallem, 2010) based on the mathematical theory developed by (Absil et al., 2009). It consists on the guessing of projection subspaces for unsampled parameter sets based on the span of already known subspace obtained by prior Model Order Reduction steps.

As one should work with computers, every subspace must be represented by a matrix that the columns span the desired space. Therefore, one must find a way to interpolate these quantities.

Direct interpolation of the matrices entries representing the subspace is obviously doomed to fail. As an example, assume a one dimensional subspace in $\mathbb{R}^n$. Any representative of this subspace is given by a vector $v \in \mathbb{R}^n$ times a non-zero scalar

$\lambda \in \mathbb{R}^*$. One expects that performing the interpolation of two equal subspaces must result in the original subspace. Taking two representatives v and $-v$ will produce the zero vector if direct interpolation of the vector entries is performed. Therefore, we have shown that this process can produce meaningless results.

The method proposed by (Amsallem and Farhat, 2008) consists in performing the interpolation in the Grassmann manifold denoted by Grass(p, n). We will not attempt to present the formal definition of manifolds and geodesics. The interested reader can find such presentations in (Spivak, 1965) and (Absil et al., 2009). We will, however, give an intuitive understanding of the method and present the formulas in a way that the the method can be readily applied.

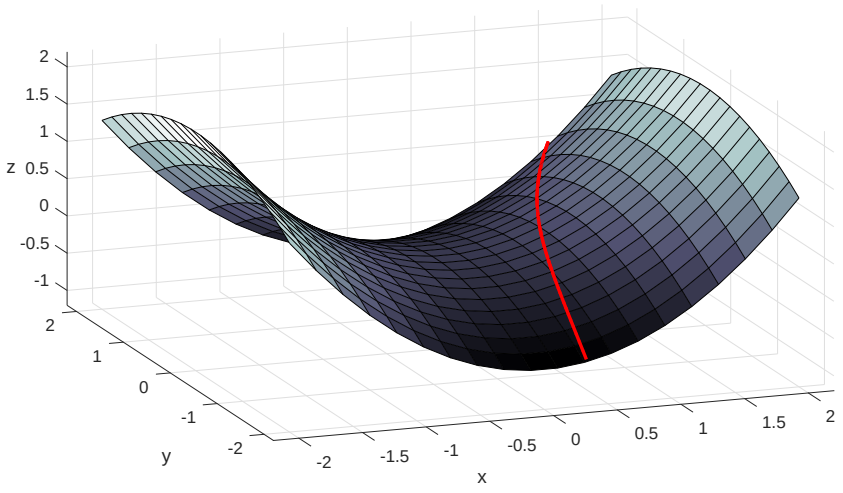


Figure 5.3: Example of a geodesic on a hyperbolic paraboloid

When attempting to interpolate the subspaces in a naive way, the result was obviously wrong because we did not respect the geometry of the underlying space were the interpolation should take place. Figure 5.3 is the depiction of an hyperbolic paraboloid. Imagine that it is the space of interest and that there are two points that lie on this surface. If we try to interpolate these two points without taking into account the curvature of this space, it is almost sure that the result will not lie on the surface. This is analogous to the failure of subspace interpolation performed in a naive way.

In the same way that straight lines are the shortest paths between two points in flat spaces, geodesics are the shortest paths between two points in a manifold. Figure 5.3 has a red line that represents one example of a geodesic on the hyperbolic paraboloid.

A meaningful interpolation schema can be obtained by tracing a geodesic between two points and taking intermediate points as the result.

One can show that a geodesic γ is described by coupled nonlinear second order differential equations (Walecka, 2007). Therefore, if the solution is unique it can be defined by one starting point and an initial velocity. One can denote the parametrized geodesic as $\gamma(t, \mathcal{V}, \widetilde{\mathcal{V}})$. We shall assume that the parameter t is contained in the interval $[0, 1]$. The quantity $\mathcal{V} \in \mathcal{M}$ is a point on the manifold, interpreted as the starting position of the geodesic. In a similar fashion, $\widetilde{\mathcal{V}} \in \mathcal{T}_{\mathcal{V}}\mathcal{M}$ is a vector at the tangent space to the manifold at $\mathcal{V}$ and can be interpreted as the initial velocity of the geodesic.

One can define two inverse maps called logarithmic and exponential maps (Absil et al., 2009). The exponential map shown in (5.22) is a function that produces the end point of the geodesic given the start point and the initial velocity. One of the most important properties of the tangent space is that it is a vector space (Lee, 2003). This means that we can perform element wise interpolation in this space as we would do in any flat space.

$$\begin{array}{rccc} \mathrm{Exp}_{\mathcal{V}_i} & : & \mathcal{T}_{\mathcal{V}_i}\mathcal{M} & \longrightarrow & \mathcal{M} \\ & & \widetilde{\mathcal{V}}_i & \longmapsto & \gamma(1, \mathcal{V}_i, \widetilde{\mathcal{V}}_i) \end{array} \tag{5.22}$$

The inverse map is defined by 5.23. Given an initial and end points, it produces the initial velocity that leads to that point.

$$\begin{array}{rccc} \mathrm{Log}_{\mathcal{V}_i} & : & \mathcal{M} & \longrightarrow & \mathcal{T}_{\mathcal{V}_i}\mathcal{M} \\ & & \mathcal{V}_j & \longmapsto & \widetilde{\mathcal{V}}_i \ : \ \gamma(1, \mathcal{V}_i, \widetilde{\mathcal{V}}_i) = \mathcal{V}_j \end{array} \tag{5.23}$$

Knowing how to perform these two maps, it is possible to perform the interpolation using a very clever schema, as shown in Algorithm 8 . In words, one start by selecting one subspace as the start point on the manifold. Using the logarithmic map one computes the velocities for each one of the other subspaces. The velocities are then interpolated using any interpolation algorithm (velocities lie on the tangent space and

Algorithm 8 Interpolation on the Grassmann Manifold

procedure $\mathcal{V}$ = INTERPOLATE($\mathcal{V}_0, \mathcal{V}_1, \ldots, \mathcal{V}_p$)
  for i = 1:p **do**
    $\widetilde{\mathcal{V}}_i \leftarrow \mathrm{Log}_{\mathcal{V}_0} \mathcal{V}_i$
  end for
  $\widetilde{\mathcal{V}} \leftarrow$ interpolate($\widetilde{\mathcal{V}}_1, \widetilde{\mathcal{V}}_2, \ldots, \widetilde{\mathcal{V}}_p$)
  $\mathcal{V} \leftarrow \mathrm{Exp}_{\mathcal{V}_0}(\widetilde{\mathcal{V}})$
  return $\mathcal{V}$
end procedure

are, therefore, vectors). Finally, an exponential map is used to obtain the interpolated point on the manifold given the interpolated velocity.

We have defined the maps but we have not shown how to compute them. The equations to perform this task are now presented. They were directly obtained from (Benner et al., 2015). Without lost of generality, we have chosen $\mathcal{V}_0$ as the start point for the geodesics. The computation of the logarithmic map requires two steps. The first one is shown in (5.24). It consists of performing the thin SVD on the quantity in the left hand side of this equation.

$$\left(\mathcal{I} - \mathcal{V}_0 \mathcal{V}_0^T\right) \mathcal{V}_k \left(\mathcal{V}_0^T V_k\right)^{-1} = U_k \Sigma_k Z_k^T \tag{5.24}$$

Taking the result of this decomposition, one applies (5.25) to obtain the velocity that is the result of the application of the logarithmic map. In this equations, note that the arctan function should be applied only to the diagonal entries of the Σ_i matrix.

$$T_k = U_k \arctan\left(\Sigma_k\right) Z_k^T \tag{5.25}$$

The exponential map also consists of two simple steps. The first one in the thin SVD of the velocity as shown in (5.26).

$$T = \hat{U}\hat{\Sigma}\hat{Z}^T \tag{5.26}$$

The point of the manifold is then given by (5.27). In a similar fashion, the sin and cos functions should be applied in the diagonal entries of the matrix.

$$\mathcal{V} = \mathcal{V}_0 \cos\left(\hat{\Sigma}\right) + \hat{U} \sin\left(\hat{\Sigma}\right) \tag{5.27}$$

As an example of application, we use the already presented cantilever beam. Two reduced order models were obtained by matching fifteen moments at the frequency zero. The first one is 1.0 meter long and the second one is 1.1 meter long. An interpolated model with length equal 1.07 m will be obtained by Subspace Interpolation. Figure 5.4 contains this result.

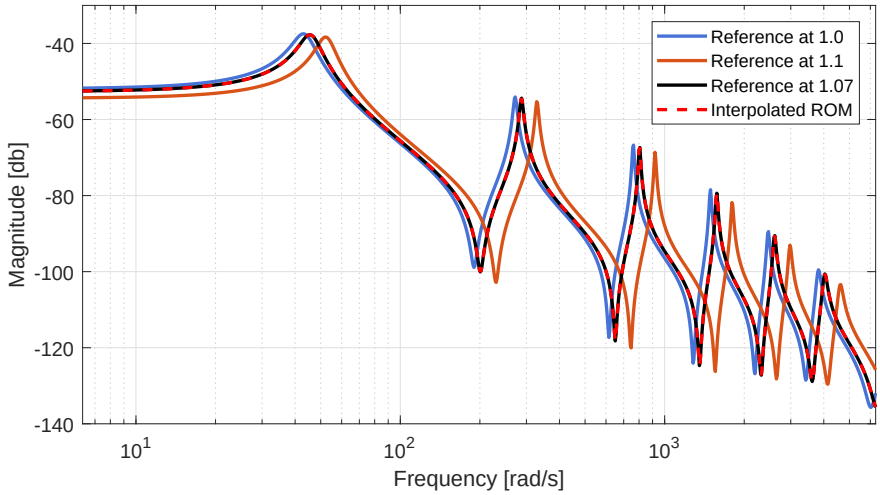


Figure 5.4: Frequency response of an interpolated Reduced Order Model by Subspace Interpolation.

This example is very instructive because it allows one to confirm that the subspace interpolation will produce a complete Reduced Order Model and not only a single scalar value as is the case in Kriging and RBF. To be able to compare the methods, we

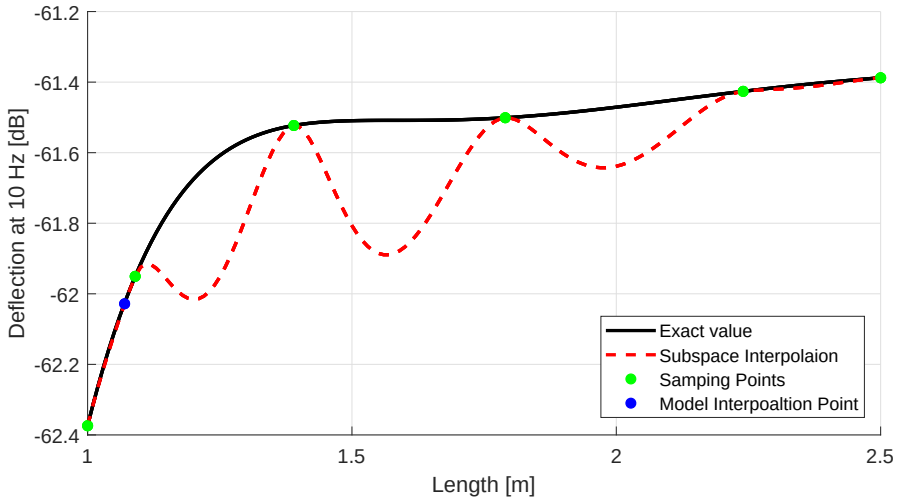


Figure 5.5: Interpolated function using Subspace Interpolation.

present Figure 5.5 with the interpolated function used in the previous examples. The point were the Reduced Order Model in Figure 5.4 has been interpolated is also shown.

We can see that the big distance between sampling points results in a poor interpolated values for the function. However, when the distance is smaller, this method is able to produce very accurately the frequency response for a wide range.

One point that must be emphasized, that will be equally true for the Matrix Interpolation technique, is the underlying hypothesis of smoothness of the state space representation when altering the parameter value. In the laminated bus bar model, due to the meshing process, this is not true. Also, the Reduced Order Models must have the same order. If this is not the case, the subspaces will not belong to the same manifold and interpolation cannot be performed in this way.

5.3.2 Matrix Interpolation

Another rather successful method to perform Parametric Model Order Reduction is called Matrix Interpolation. It interpolates the resulting matrices for different Reduced Order Models to obtain the equations for an unsampled set of parameters.

It should be clear that direct interpolation without any prior treatment of the matrices will result in meaningless results, once that, as shown in Chapter 2, any system in

state space form can be written in an equivalent form by means of a similarity transformation. Interpolating the entries of two equivalent systems may result in nonsense if they are not represented by the same base.

Therefore, one should represent the systems in a similar base and only then performing interpolation. Once this is done, the new Reduced Order Model can be obtained as shown in (5.28).

$$\begin{cases} \widetilde{E}_r \dot{x}_r &= \widetilde{A}_r x_r + \widetilde{B}_r u \\ y_r &= \widetilde{C}_r x_r \end{cases} \tag{5.28}$$

Where the matrices $\widetilde{A}_r$, $\widetilde{B}_r$, $\widetilde{C}_r$ and $\widetilde{E}_r$ are the entrywise interpolated matrices, given, as an example by (5.29). Each weight ω_i is a function of the desired parameter vector and the matrix $A_{r,i}$ is the one associated with the i-th reduced order model.

$$\widetilde{A} = \sum_i \omega_i(\hat{p})\, A_{r,i} \tag{5.29}$$

Any classic interpolation algorithm can be used to perform the interpolation process. Particularly, we have used a simple strategy that has empirically shown to work well. If some parameter vectors $p_1, p_2, \ldots, p_n$ have been sampled, one can use weights of their convex combination as interpolation weights. The linear system to be solved is shown in (5.30).

$$\begin{bmatrix} p_1 & p_2 & \cdots & p_n \\ 1 & 1 & \ldots & 1 \end{bmatrix} \begin{bmatrix} \omega \end{bmatrix} = \begin{bmatrix} \hat{p} \\ 1 \end{bmatrix} \tag{5.30}$$

This interpolation schema will give exact results for systems whose parameters affect linearly the matrices entries. If the user has any previous knowledge of specific

nonlinearities, they can embed this knowledge in the system by applying a transformation to the rows of the matrix in a manner that the transformed parameters obey this linear equation.

There are different methods to bring the Reduced Order Models into an appropriated form that allows interpolation (Benner et al., 2015). Here, however, we concentrate our attention to the one described by (Panzer et al., 2010).

The method starts by forming a matrix U as in (5.31).

$$\mathcal{V} = [\ \omega(p_1)\mathcal{V}_1\ \omega(p_2)\mathcal{V}_2\ \ldots\ \omega(p_n)\mathcal{V}_n\ ] \tag{5.31}$$

One performs the SVD of $\mathcal{V} = U\Sigma Z^T$ and keep only the r first columns of U, denoted by $\widetilde{U}$. The matrices N_i and M_i in (5.32) are built to perform the transformation that brings the systems to an appropriate base.

$$N_i = (\mathcal{V}_i^T\widetilde{U})^{-1} \qquad M_i = (\mathcal{W}_i^T\widetilde{U})^{-1} \tag{5.32}$$

Explicitly, the transformed system is obtained as in (5.33).

$$\begin{cases} M_i\widetilde{E}_{r,i}N_i\dot{x}_r &= M_i\widetilde{A}_{r,i}N_ix_r + M_i\widetilde{B}_{r,i}u \\ y_r &= \widetilde{C}_{r,i}N_ix_r \end{cases} \tag{5.33}$$

As an example of application, we use exactly the same case for subspace interpolation. We pick two models, one having 1.0 meter and the other with 1.1 meter and interpolate the system at 1.07 meters. Figure 5.6 contains this result.

For the sake of comparison, we present the function used for Kriging and RBF. The result can be seen in Figure 5.7. The interpolation point corresponding to the result presented in Figure 5.6 is also shown.

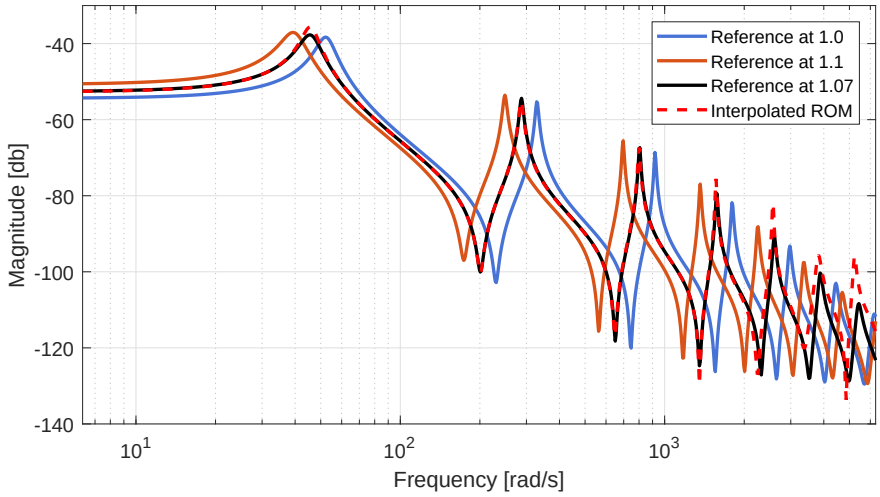


Figure 5.6: Frequency response of an Interpolated Reduced Order Model by Matrix Interpolation

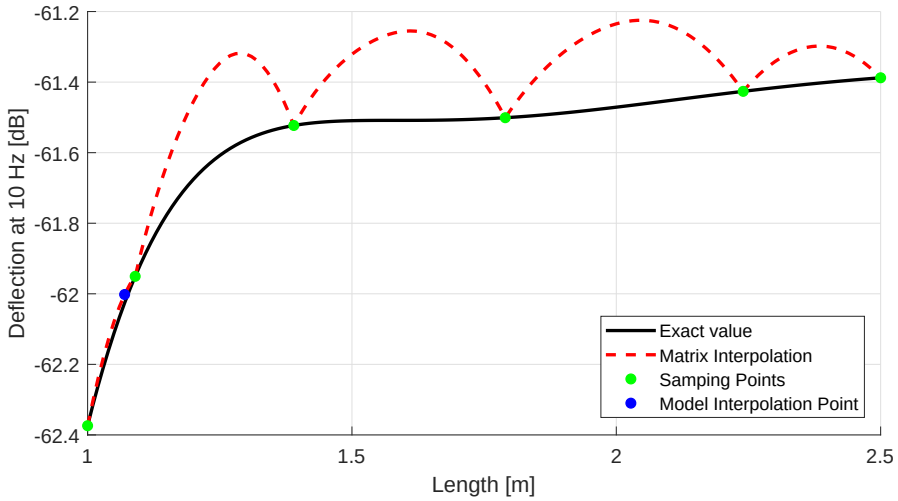


Figure 5.7: Interpolated function using Matrix Interpolation.

The result is very similar to the one obtained for Subspace Interpolation. Although the method has been shown to work well to produce reduced order models, it was not as efficient in interpolating an objective function when the sampling points are far apart. The interpolation, however, is very fast once that it is performed on the matrices of the Reduced Order Models.

5.4 Conclusion

In this chapter we have presented four methods that were divided in two big classes. Each one with specific advantages and caveats.

Radial Basis Function interpolation is very simple and has a straight forward implementation. We have shown that, if the parameters are chosen correctly, it can produce accurate results. Kriging is much more complex method that, besides producing accurate results, return the confidence interval of the interpolated value. This will be extensively used in the following chapter.

These two methods are only able to produce scalar results. Therefore, if one seeks a complete dynamic model, different approaches must be employed. Subspace Interpolation is a very elegant and powerful method to produce Interpolated Reduced Order Models. Differently from RBF and Kriging, this method produces models that can be used for simulation in a large frequency range or directly connected to external routines. However, we have seen that it does not perform so well if the sampling points are chosen too far apart. In a similar manner, Matrix Interpolation is also a method that is capable of producing Interpolated Reduced Order Models. It is conceptually simpler than Subspace Interpolation and has presented the same problem concerning very distant sampling points.

Subspace Interpolation and Matrix Interpolation have the disadvantage of assuming state space representations that are smooth in respect to the parameter. Also, the dimension of the Reduced Order Models must be the same. Kriging and RBF, however, only deal with values and are not constrained by such restrictions.

The next chapter will present different manners to use these methods to perform optimization using Reduced Order Models.

Chapter 6

Application of Model Order Reduction in Optimization

6.1 Introduction

In this chapter we present some applications of Reduced Order Models in optimization. The main focus is the coupling of the Subspace Interpolation Method and Kriging, presented in Chapter 5, with an optimization algorithm that takes advantage of them.

We also present two optimization problems. One consists in the positioning of wires inside a cage in order to minimize their mutual impedance. The second one uses the laminated bus presented in Chapter 4. The problem can be basically described by the optimal placing of capacitors on this system in a way that the current that passes through them is well balanced. This will prevent the premature burnout of one of them reducing the lifespan of the system as a whole.

The following sections present in detail the algorithms that were used in the final application section.

6.2 Optimization Algorithm

The literature in optimization contains a large variety of algorithms that are very well understood and have been shown to perform well in practice. Some of the most well known are the simplex, steepest descent, Newton-Like, genetic algorithm, branch and bound, among other methods (Fletcher, 2013), (Rao, 2009). Each one of them will be best suited for a particular kind of problem: convex, linear, nonlinear,

constrained/unconstrained, mono-objective/multi-objective, etc. In this chapter we concentrate on optimization of nonlinear functions.

The above listed algorithms work with the hypothesis that there is a function that performs accurate computation of an objective function. Performing optimization with surrogate or Reduced Order Models is different from directly working with the High-Fidelity Model that may be used to directly compute the values for this function. Therefore, the following sections describe an algorithm that is designed specifically to deal with these differences.

6.2.1 Optimization with Surrogate Models

When using a High-Fidelity Model to perform optimization, one assumes that the computation of the objective function can be carried out for a number of needed points in the design space. If this is true, there are many optimization algorithms that can be employed. The problem arises when the computational burden to perform these calculations becomes prohibitive, as described in (Antoulas, 2005).

When one uses surrogate models to accelerate computations, they have to face a compromise between speed and accuracy. Therefore, the simulation response may be obtained much quicker, but the results may be inaccurate. There are, however, many different strategies that allows the use of these models in optimization in a reliable way.

One common feature among many optimization algorithms using surrogate models is the sampling of the search space (Jones et al., 1998), (Knowles, 2006). This allows the construction of a first response surface that approximates the behavior of the objective function all over the design region. If there are enough and well chosen sampling spots, this surface can be generated only once and used as an approximation for the optimization function during the whole optimization process. Most of the algorithms, however, will be based on some sort of adaptive sampling of the design space. Therefore, the surface will be refined and recomputed at each inner cycle of the algorithm.

There are different and well known sampling strategies in the literature. We focus on three: Random Sampling, Full Factorial and Latin Hypercube (Oehlert, 2010) (Montgomery, 2017).

The random sampling strategy, consists in simply picking points in the design space at random. The problem with this strategy is that is can produce points that

are clustered in some region of the space. As we have seen in the presentation of the Kriging algorithm, one expect nearby points to have high correlation and consequently, not much new information. From a numerical point of view, this could result in poorly conditioned matrices (Ball, 1992).

One way of spreading the points evenly in space that may work for low dimensional problems is known as full factorial design. It consists at placing points evenly spaced on a grid in n dimensions. If we assume that every dimension is partitioned in d divisions, then the number of sampling points is given by $(d+1)^n$. This shows why this is an impractical method for high dimensional problems.

One strategy that is somehow in the middle of the previous two and which is very often used in practice is the Hypercube Latin Sampling (Oehlert, 2010). It consists on the division of space in a grid in dimension n and placing sampling points at random inside each one of the squares of the grid. There is an additional constraint that the projection of the occupied squares on the coordinate axes shall not overlap. An intuitive picture of this placement can be obtained by imagining a chess board in n dimensions where one must place a given number of rocks, guaranteeing that they are not attacking each other.

Figure 6.1 contains an illustration of the three different sampling methods. It is inspired in a similar illustration found in (Baudoui, 2012).

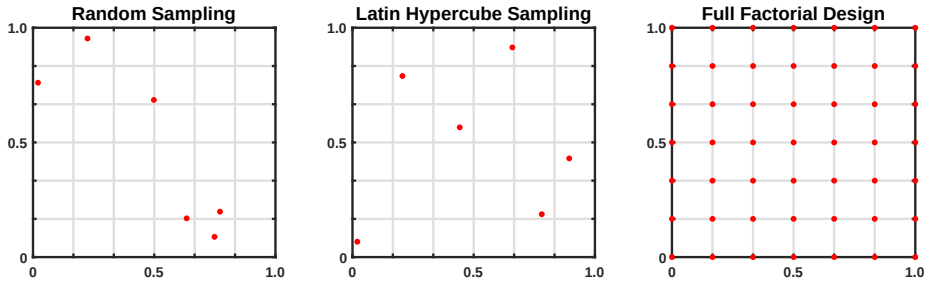


Figure 6.1: Illustration of the different sampling methods

Having performed the sampling of the search space, one may perform optimization on the surrogate model constructed by using one of the techniques that have been presented in Chapter 5 (Kriging Estimation, Radial Basis Interpolation, Subspace Interpolations and Matrix Interpolation). However, the initial sampling may be refined during the optimization process. The following paragraphs describe a way of doing so that is based on the Efficient Global Optimization (EGO) algorithm (Jones et al., 1998).

After applying the sampled strategy, one has a sample set $\mathcal{S} \subset \mathbb{R}^n \times \mathbb{R}$ containing $n_s \in \mathbb{N}$ sampled points. The algorithm looks for the minimum value of objective function among all the samples and assigns the point related to it as the current solution. If one represents these coordinates in the design space by $x^* \in \mathbb{R}^n$ and the corresponding objective function as $y^* \in \mathbb{R}$, then this solution can be described by the ordered pair (x^*, y^*).

A mechanism to refine the sampling is then applied. It consists in adding $n_i \in \mathbb{R}$ (an user defined parameter) points per iteration to $\mathcal{S}$ having as criterion a variation of the Expected Improvement (EI). In its original form, it is a non-smooth function whose maximization can be performed using a branch and bound strategy (Jones et al., 1998). We have chosen to use the lower bound of the 95% confidence interval given by the Kriging estimator (6.1) as the criterion to refine the grid. It has the advantage of being a smooth function. An internal Genetic Algorithm is used to choose the points. Note that σ denotes the standard deviation of the error at a point given as its parameter. The value y^i refers to the inferior lower bound on the 95% confidence interval and $y^s(x^s)$ and x^s are objective function values and sampling points, respectively.

$$y^i = y^s(x^s) - 2\sigma(x^s) \tag{6.1}$$

This process is repeated until the maximum number of iterations supplied by the user is reached. Once that the response surface has been refined, an optimization procedure is launched to find the minimum predicted value. If this point is not already contained in $\mathcal{S}$, then, it is added to the surface. At this point the new best solution (x^*, y^*) may be fond. It is very important to note that the fist refining process is performed using the Expected Improvement as an objective function and the second one uses the predicted value without any further processing. In order to illustrate this procedure in a concise manner, Algorithm 9 contains its pseudo-code.

In the next section, we present the use of this method to perform the optimization of an electromagnetic system.

Algorithm 9 Optimization with Surrogate Models

procedure (x^*, y^*) = OPTIMIZE(n_s, n_i, *nIter*)

  $\mathcal{S} \longleftarrow$ Latin Hypercube Sampling (n_s)
  $\mathcal{M} \longleftarrow$ Build Model($\mathcal{S}$)
  $(x^*, y^*) \longleftarrow$ minimum($\mathcal{S}$)

  $i \longleftarrow 0$
  while $i \leq nIter$ **do**
    for j = 1,..., n_i **do**
      $(x^s, y^s) \longleftarrow$ Genetic Algorithm ($\mathcal{M}$, EI) ▷ Expected Improvement
      $y^H \longleftarrow$ Evaluate HFM (x^s)
      $\mathcal{S} \longleftarrow \{\mathcal{S}, (x^s, y^H)\}$
      $\mathcal{M} \longleftarrow$ Update Model($\mathcal{S}$)
    end for

    $(x^s, y^s) \longleftarrow$ Genetic Algorithm ($\mathcal{M}$) ▷ Surrogate Model Predictions
    $y^H \longleftarrow$ Evaluate HFM (x^s)
    $\mathcal{S} \longleftarrow \{\mathcal{S}, (x^s, y^H)\}$
    $\mathcal{M} \longleftarrow$ Update Model($\mathcal{S}$)

    if $y^H < y^*$ **then**
      $(x^*, y^*) \longleftarrow (x^s, y^H)$
    end if

    $i \longleftarrow i + 1$
  end while

  return (x^*, y^*)
end procedure

6.3 The Problem of Placing Capacitors at a Power Bus

To illustrate the strategy that uses Kriging and the variation of the EGO algorithm, we have adapted the system first introduced in Chapter 4 to create an optimization problem. We have reproduced its representation in Figure 6.2

It consists in finding the position of the capacitors that minimizes the current difference among then in the whole frequency range of interest. The idea behind this is that if one of them gets consistently more current than the others then it will probably fail prematurely reducing the estimated lifespan of the system.

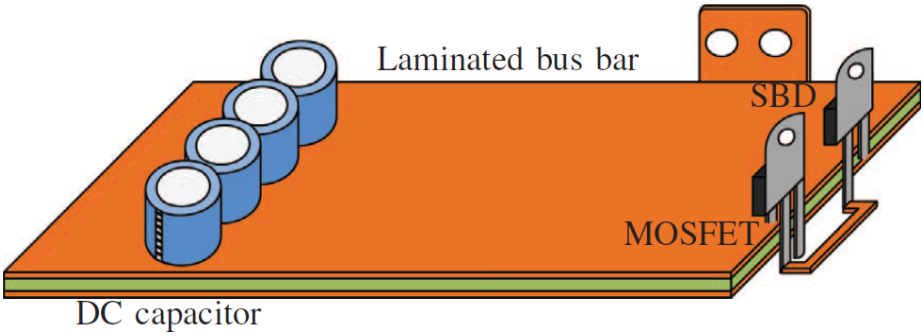


Figure 6.2: Drawing of the laminated bus bar. Adapted from the original (Kuwabara et al., 2016).

Therefore, we would like to minimize the current differences between the capacitors for the frequency range of operation. If we denote the mean values of the currents (obtained by integration) by $m \in \mathbb{R}^n$, then a suitable objective function would be given by (6.2).

$$f(m) = \sum_{i=1}^{4} \sum_{j>i}^{4} (m_i - m_j).^2 \tag{6.2}$$

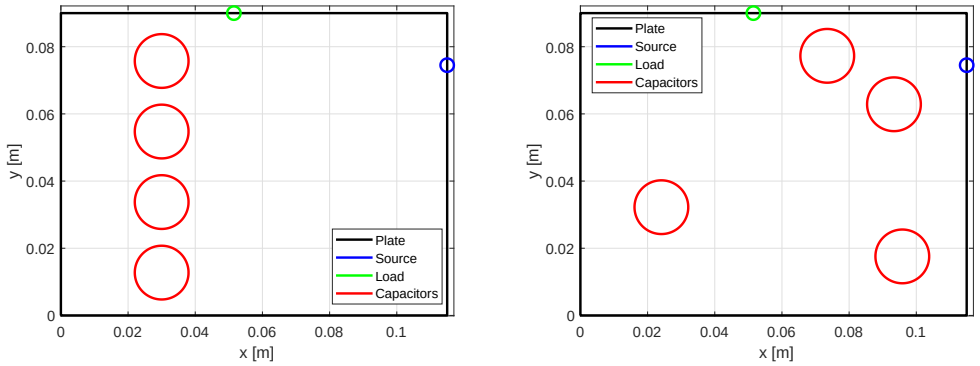
One obvious solution that solves this problem is placing every single capacitor at the same point. This however is not practical due to the fabrication issue. A penalty term is then added to the objective function to avoid this unpractical situation (Rao, 2009). Its expression can be seen in (6.3). The parameter ρ is a penalty term and r is the radius of the capacitor. For the given problem, we have chosen ρ fixed at 5×10^3. This may seem exaggerated at a first analysis, however the distance in computed in millimeters and this value acts as a conversion factor to bring the quantities closer together.

$$p(x,y) = \rho \sum_{i=1}^{4} \sum_{j>i}^{4} \max\left\{0,\ 2r - \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}\right\} \tag{6.3}$$

The box constraints of the problem are also chosen so that no capacitor is too close to the edge of any plate. We have chosen the minimal distance as being 4 mm. This

is not arbitrary. This value corresponds to the precision that the number of cell grids used to discretize the space allow us (32×25).

The original placement and the optimization result is illustrated in Figure 6.3. The optimization process took 3 hours and 3 minutes, being 42 minutes spent in the process of the initial sampling. A total of 40 generations was used with 3 new points at each iteration. The Latin Hypercube Sampling was performed with 80 points, being a total of 10 per each variables.



(a) Original position (b) Optimized position

Figure 6.3: Comparison between the original and optimized position for the capacitors.

It is natural to wonder about the correctness of the obtained solution. To confirm that the answer is plausible, we show the voltage distribution for the original and optimized positions in Figures 6.4a and 6.4c, respectively.

Sampling the voltage is easy because it is the unknown variables for capacitors in the modified nodal analysis. To transform this to current, one could simply apply the definition of impedance. After this process, the current in the original and optimized configurations can be seen in Figures 6.4b and 6.4d, respectively.

Note that the voltages and currents are much more well distributed over the capacitors for the optimization results in Figure 6.3. There is still a small frequency range where they are unequal, but the overall result is better.

During the optimization process no direct call to the High-Fidelity Model was made. Every single sampling of the objective function was done by applying a reduction step and computing on the Reduced Order Model. The accuracy of the result is guaranteed by the adaptive expansion point placement developed in this thesis at Chapter 3.

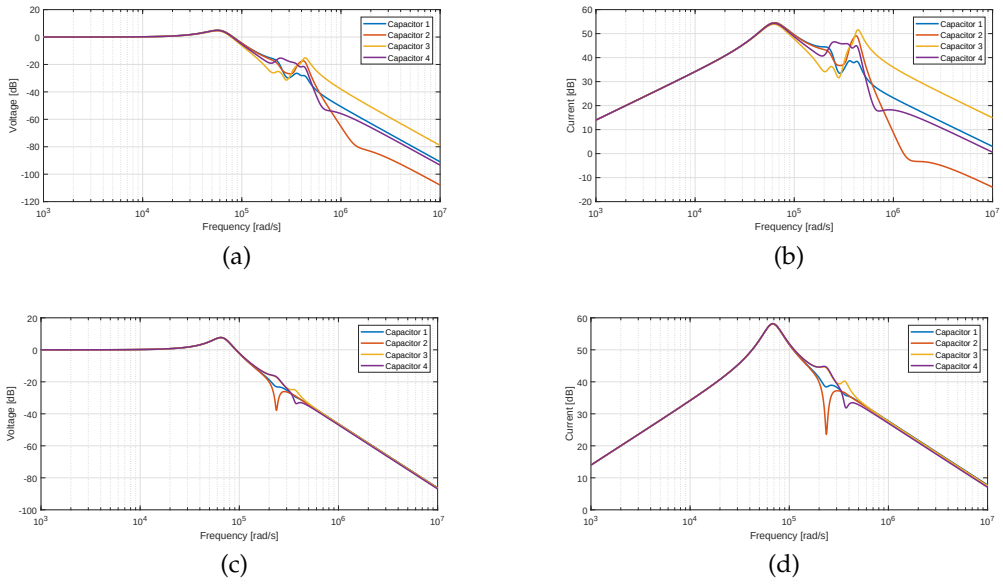


Figure 6.4: Optimization results. (a) Voltage on the original system. (b) Current on the original system. (c) Voltage on the optimized system. (d) Current on the optimized system.

Before preceding to the introduction of another optimization strategy, we explain why this model has been optimized using the Kriging interpolation procedure and not the Subspace Interpolation one. As we have stated before, the limited size of the grid only allows a precision for the placement of the capacitors of about 4 mm. Therefore, if the optimization algorithm tries to move the capacitor by a value lower than this threshold, it would not change the state space. The capacitor would be connected to the same electrical node of the system. Therefore, the state space that represents this model is not smooth when it is threated as a function of the capacitor positions. Therefore, interpolation methods like Matrix Interpolation and Subspace Interpolation are not applicable to this situation.

6.4 The Simplex Sampling Strategy

In this section we present an alternative kind of sampling that we have developed called the Simplex Sampling Strategy (AO Leite et al., 2016). Its application aims at Newton based algorithms for single objective optimization problems. These kind of algorithms will produce a sequence of candidate solutions that should eventually converge into the minimum point (Fletcher, 2013). As a canonical example, one can

think of the steepest descent method. These algorithms will normally follow a well defined path. Therefore, there are parts of the design space that will never be analyzed. The simplex sampling strategy takes advantage of this fact and will iteratively add points in a way that only the regions of interest to the optimization algorithm are sampled.

Algorithm 10 presents the complete strategy. Note that to be able to implement the method one need to successfully realize three processes. The first one is the construction of a simplex in arbitrary dimension around a given initial point. The second one is to distinguish between points inside and outside a simplex. Finally the third one is the capacity to reflect the simplex around a given reflection point.

Algorithm 10 Optimization with Simplex Sampling Strategy

```
1: procedure OPTIMIZE(x)
2:     S ← Build simplex around x
3:     while optimization not converged do
4:         x ← advance point (x)
5:         while x not inside S do
6:             S ← Reflect Simplex (S, x)
7:         end while
8:     end while
9: end procedure
```

We start by showing how to construct a regular simplex in arbitrary dimension. Given $n+1$ normalized vectors $v_i \in \mathbb{R}^n$, if one identifies the points of the simplex with the values of these vectors when placed at the origin, then property (6.4) defines the simplex. Note that as the vectors are unitary, the cosine between any two of them $\cos(\alpha)$ is given by $a \in \mathbb{R}$.

$$v_i^T v_j = \begin{cases} 1 & \text{if} \quad i = j \\ a & \text{if} \quad i \neq j \end{cases} \tag{6.4}$$

As the number of vectors exceeds the dimension of the space by one unity and

we assume that there are no collinear vectors, then it is guaranteed the existence of non-zero coefficients α_i such that (6.5) is possible.

$$\alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_n v_n + \alpha_{n+1} v_{n+1} = 0 \tag{6.5}$$

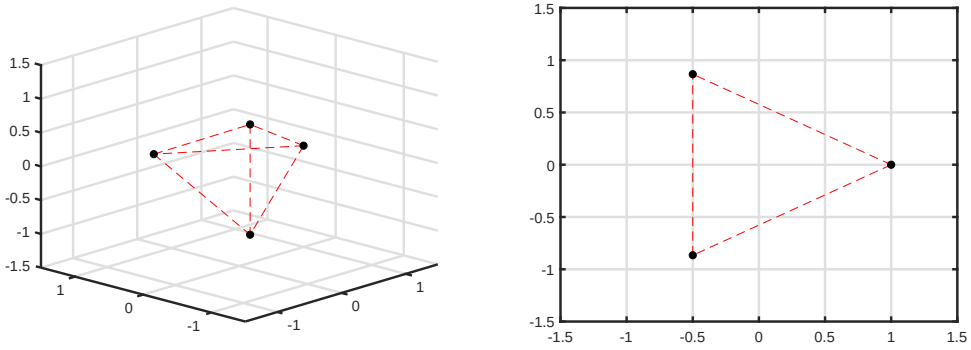
Multiplying (6.5) by each one of the vectors and arranging the result in matrix form given rise to (6.6).

$$\begin{bmatrix} 1 & a & a & \dots & & \\ a & 1 & a & \dots & & \\ a & a & 1 & \dots & & \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ & & & \dots & 1 & a \\ & & & \dots & a & 1 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \vdots \\ \alpha_n \\ \alpha_{n+1} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} \tag{6.6}$$

For an arbitrary value of a the null space of this matrix is empty (set $a = 0$ for a trivial case). However, the existence of non zero values for α_i is assured by the definition of linear dependency (Strang, 2011). Therefore, (6.6) forces the conclusion that $a = -1/n$. One can see by inspection that if this is the case, then the vectors of all ones is in the null space of the matrix.

Having established this property, one can generate simplexes of arbitrary dimension using the procedure described in (Conn et al., 2009). It consists in forming a matrix V by placing side-by-side the first v_i vectors of the simplex (up to this point they are still unknown). Defining a new matrix $C = V^T V$, its entries can be computed by direct application of (6.4). Due to the Gershgorin circle theorem (Gershgorin, 1931), this matrix is positive definite and therefore has a corresponding Cholesky decomposition $C = LL^T$. The first n vectors composing the simplex are then chosen as being the columns of L^T. The last one can be directly computed by (6.5), knowing that this equality is satisfied when all the values of α_i are unitary. One can verify that these values satisfy the definition of a simplex by direct computation of the inner products.

Arbitrary simplexes can then easily be obtained by rotation and translation of the points. Figure 6.5 shows examples of simplexes in two and three dimensions obtained by this method.



(a) Simplex in 3D (b) Simplex in 2D

Figure 6.5: Examples of simplexes obtained by the generating method.

The interior of the simplex is defined as the convex combination of all its points. Formally it is given by (6.7). In this equations the values of α_i are nonnegative.

$$I = \left\{ x \in \mathbb{R}^n \;\middle|\; x = \sum_{i=1}^{n+1} \alpha_i v_i \,,\; \sum_{i=1}^{n+1} \alpha_i = 1 \right\} \tag{6.7}$$

One direct way of detecting if the point is in the interior of the simplex is to compute the values for the coefficients α_i and verify if they all are in the range $0 \leq \alpha_i \leq 1$. Given a target point c, these coefficients are the solutions of the linear equations (6.8).

$$\left[\begin{array}{cccc} v_1 & v_2 & \dots & v_{n+1} \\ \hline 1 & 1 & \dots & 1 \end{array}\right] \left[\begin{array}{c} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_{n+1} \end{array}\right] = \left[\begin{array}{c} c \\ \hline 1 \end{array}\right] \tag{6.8}$$

Note that if one of the values is zero this means that the target point c is exactly on one of the faces. In practice however, this is extremely unlikely and this case will be ignored in the following analysis.

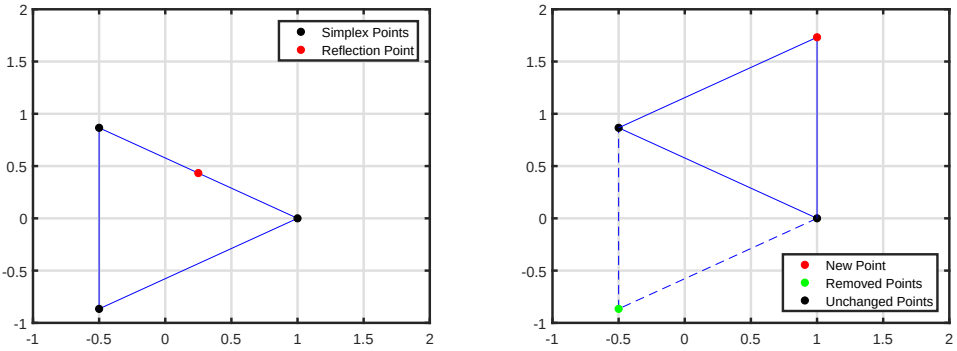
When the optimization point is outside the simplex, new samplings must be performed until it is inside the geometric figure again. The sampling points are chosen by means of reflections of the farthest point of the simplex from the current optimization point.

Setting the index j as the one that corresponds to the coefficient with biggest negative value in modulus, one can define a reflection point r as in (6.9).

$$r = \frac{1}{n} \sum_{\substack{i=1 \\ i \neq j}}^{n+1} v_i \tag{6.9}$$

The vector r is the result of a linear combination of the vectors v_i with one zero coefficient. Therefore, it is on one face of the simplex. The reflected point v_r is defined by (6.10). Figure 6.6 shows one example of such reflection.

$$v_r = v_j + 2(r - v_j) \tag{6.10}$$



(a) Identification of Reflection Point (b) Reflected Simplex

Figure 6.6: Example of reflected simplex.

The simplex will then follow the optimization point creating samples around it at any iteration. Figure 6.7 contains an illustration of the main idea behind the use of this sampling strategy and interpolation of the objective function.

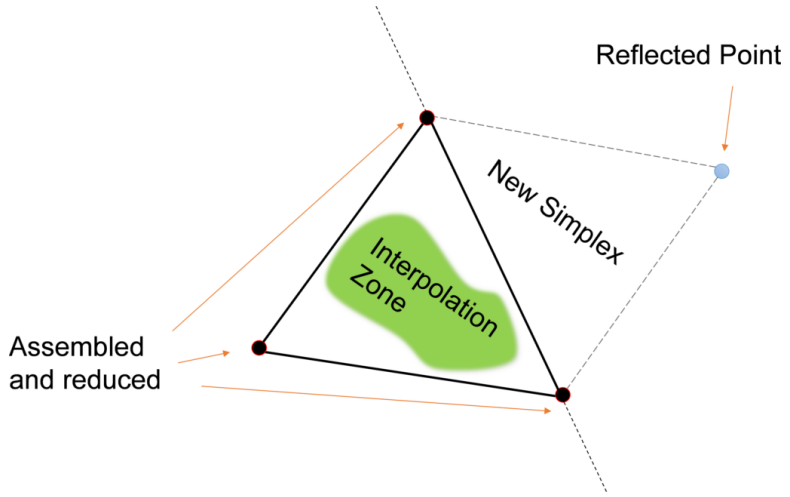


Figure 6.7: Interpolation zone inside a simplex

In the following section we present an example of application that uses the simplex sampling strategy in combination with the subspace interpolation technique.

6.5 The Problem of Placing Wires in a Cage

To illustrate the simplex sampling strategy we present a simple but instructive example. It consists in placing wires inside a metallic cage in order to maximize the impedance among the wires for a wide frequency range, minimizing interference. As it is a two-dimensional problem, it allows the visualization of the process. Figure 6.8 contains the geometry of the system. The wires are moved together in order to restraint the number of variables of the problem.

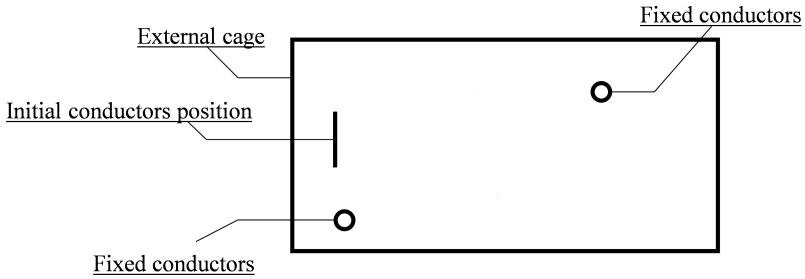


Figure 6.8: Illustration of the system.

To solve this problem, we have used the simplex sampling strategy in combination with the subspace interpolation presented in Chapter 5. The optimization algorithm is a

classic Quasi-Newton using the BFGS update formula for the Hessian matrix (Fletcher, 2013). We have chosen to use the mean value of the impedance as the objective function. As the integral and the mean value are related by a constant, in the actual implementation, we minimize the integral of the impedance.

The problem was solved in three different manners. The first one ignores the possibility of working with Reduced Order Models and performs the optimization with the High-Fidelity Model directly. The second one uses reduction by moment matching with adaptive expansion point placements to compute the objective function at each optimization step. Finally, the third one perform the reduction and interpolation of the model using subspace interpolation techniques in each simplex, during optimization. The result of these processes can be seen in Figure 6.9.

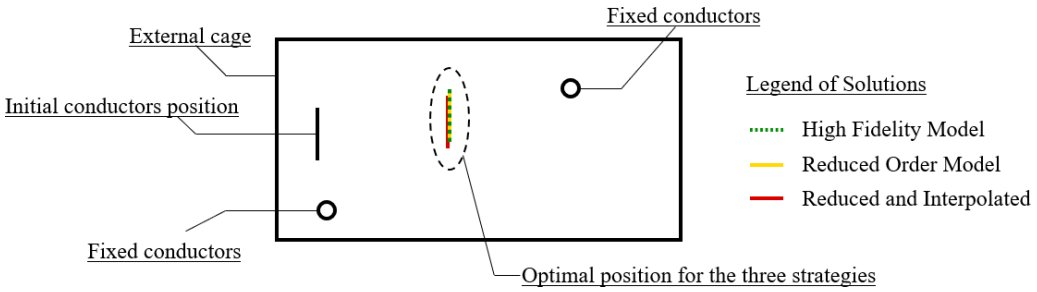


Figure 6.9: Optimization results.

The number of calls to each type of function evaluation can be seen in Table 6.1. The time and objective function values obtained are listed in Table 6.2. The High-Fidelity Model has order 1765 and the Reduced Order Models have been fixed to have order 40, allowing to perform the Subspace Interpolation.

	Model Reductions	Model Interpolations	HFM Calls
HFM	0	0	36
ROM only	36	0	0
ROM and Interpolation	8	28	0

Table 6.1: Number of calls by type of optimization process.

It is important to note that the High-Fidelity Model and the Reduced Order one have very close objective functions. This illustrates the success of the adaptive expansion point strategies introduced in Chapter 3.

It is very instructive to see how the simplex moved during the optimization process. This result can be seen in Figure 6.10. Note that as the evolution occurs in small enough

	X [m]	Y [m]	Objective Function [dB Hz]	Time [s]
HFM	0.4230	0.2431	-55.5542	624
ROM only	0.4232	0.2431	-55.5543	163
ROM and Interpolation	0,4195	0,2282	-55.5611	65

Table 6.2: Results for the different optimization approaches.

steps, each reflection of the simplex was capable of capturing the points outside of the interpolation zone.

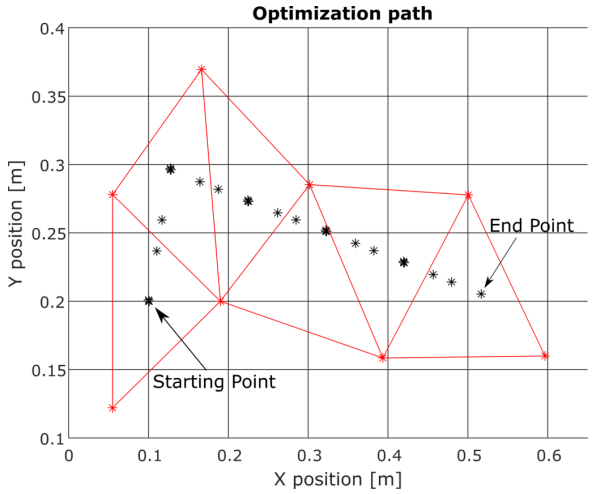
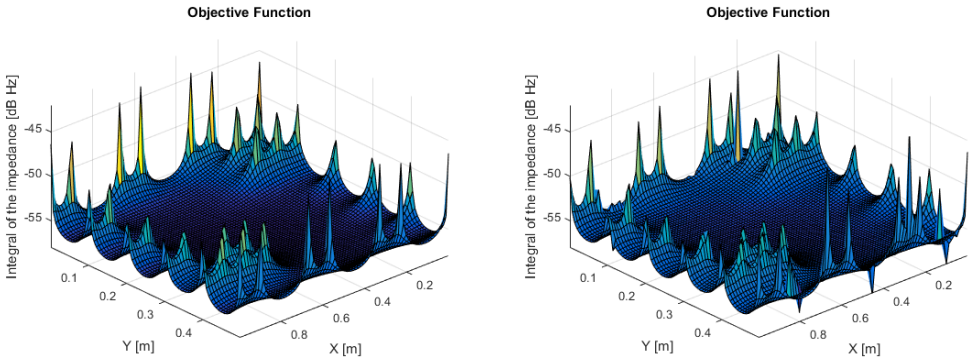


Figure 6.10: Simplexes obtained during the evolution of the optimization point

As the problem has only two dimensions, we can create a picture with the landscape of the objective function for the High-Fidelity Model and the reduced one. This is explicitly shown in Figure 6.11. Of course in the real optimization process we would never construct this surface. It is here only for the appreciation of how well the interpolation method is capable of reproducing the objective function.

Having all these informations, one can conclude that the simplex sampling strategy combined with the subspace interpolation technique has produced very accurate results ten times faster than the classic approach.

We have been able to use this method because as there are no external discrete components, the state space that represents this model is a smooth function of the parameters and can be interpolated using the Subspace Interpolation technique.



(a) High-Fidelity objective function (b) Interpolated objective function

Figure 6.11: Comparison between the original and interpolated objective function.

6.6 Conclusion

In this chapter we have presented algorithms capable of performing optimization using surrogate models. The results were presented in two different applications. One of them used a classic Quasi-Newton algorithm coupled with the simplex sampling strategy developed by the authors and the other a variation of the EGO algorithm.

The results are very satisfying and the algorithms were capable of producing very accurate solutions in a much shorter time than would be spent by conventional methods.

Each method has been employed in an specific situation. The Subspace Interpolation technique depends on a state space that is a smooth function of the parameters. Therefore, Kriging has been the choice for optimization in the laminated bus bar case. The second optimization system has been designed in a way that this restriction is satisfied, allowing the use of the Subspace Interpolation technique.

We conclude by saying that Model Order Reduction coupled with surrogate techniques for optimization can produce useful results.

Chapter 7

Conclusion

We have presented the problem of simulation and optimization of electromagnetic devices using Model Order Reduction as a tool to reduce the computational burden.

The first chapters were devoted to explaining the algorithms and mathematics behind this complex subject. Many examples were presented and demonstrated that Model Order Reduction is a powerful technique.

The problem of time domain simulation leads to the conclusion that direct application of classic methods was not enough to perform this task accurately. Therefore, we have developed techniques that allow the production of accurate and stable Reduced Order Models.

These modification in the algorithms were tested in two complex electromagnetic systems and showed to perform well in these cases. Therefore, one can use the fast and accurate Reduced Order Model instead of original one, substantially reducing the computational burden of simulation processes.

The second part of the text presented the problem of performing optimization using very complex models whose simulation is time consuming. A second time, Model Order Reduction was employed to reduce the computational burden.

Different algorithms were presented capable of coupling order reduction with surrogate modeling. They have been shown to produce satisfying results, as was shown in two different optimization problems.

We have been able to produce fast, accurate, and stable models that present a very fell order when compared with the High-Fidelity model. Therefore, we can conclude that the techniques developed in this thesis are very well suited to performing simulation and optimization of large scale systems.

Following this work, we would like to apply the Model Order Reduction Framework and associated optimization techniques to a more realistic mammography scenario. This would require the use of optimization in order to solve an inverse problem. We believe that this technique has great potential and can be used to significantly simplify the challenges involved in the process.

Nonlinear models have not been dealt with in this thesis. Therefore, this is one important point to be investigated at the future. A generalization of the Moment Matching approach for this class of problems would be very welcome.

Finally, we have not threated optimization problems with a very large number of parameters. We have concentrated our attention to valuer smaller than 10. It would be important to understand the behavior of algorithms for these situations and identify improvements that must be done so that this number could be scaled to the hundreds.

Appendix A

Updates of the Generalized Eigenvalue Problem

This appendix is divided into three main sections. The first one shows a particular rank-1 update to the generalized eigenvalue problem that is capable of changing the value of a specific eigenvalue without changing the other ones. In this presentation, the proof used to obtain a relation between the determinants of two matrices related by a rank-1 update is developed. It is longer than the classical one, but can be obtained by direct application of linear algebra concepts.

The second main section will generalize this idea to the case of an update of arbitrary rank. It would also be possible to inverse the order of the presentation and present the rank-1 update as a particular case of the rank-k update. However, this approach would not be didactic and would not be very instructive.

Finally, it is shown that for the case of a rank-2 update it is possible to change simultaneously the value of a pair of complex conjugate eigenvalues and keep the problem in the field of the real numbers.

A.1 Rank-1 Update of the Generalized Eigenvalue Problem

The main objective of this section is to show how the generalized eigenvalue problem changes when some very special rank-1 updates are performed. The starting point is the generalized eigenvalue problem shown in A.1.

$$Mu = \lambda Nu \tag{A.1}$$

The updated matrices are defined by (A.2). Notice that the vector u in the rank-1 update is the eigenvector of the eigenvalue problem. This definition is shown before its actual use because they will make the following development much simpler.

$$\begin{cases} M_u = M + Nuv^* \\ N_u = N + Nuw^* \end{cases} \tag{A.2}$$

At this point we will add and subtract the same quantities to the matrices M and N, leaving the generalized eigenvalue problem intact. This and the immediate development of this expression can be seen in (A.3). The terms in boldface where added equally in the two sides of the equations.

$$\begin{aligned}
Mu &= \lambda Nu \\
(M + Nuv^* - Nuv^*)u &= \lambda(N + Nuw^* - Nuw^*)u \\
(M_u - Nuv^*)u &= \lambda(N_u - Nuw^*)u \\
M_u u + \lambda Nuw^*u &= \lambda N_u u + Nuv^*u \\
M_u u + Muw^*u + \boldsymbol{Nuv^*uw^*u} &= \lambda N_u u + Nuv^*u + \boldsymbol{Nuv^*uw^*u} \\
M_u u + (M + Nuv^*)uw^*u &= \lambda N_u u + (N + Nuw^*)uv^*u \\
M_u u + M_u uw^*u &= \lambda N_u u + N_u uv^*u \\
M_u u(1 + w^*u) &= N_u u(\lambda + v^*u) \\
M_u u &= \tfrac{\lambda + v^*u}{1 + w^*u} N_u u
\end{aligned} \tag{A.3}$$

The last line of this equation is so important that it will be put in a separate Equation (A.4).

$$M_u u = \frac{\lambda + v^*u}{1 + w^*u} N_u u = \lambda_u N_u u \tag{A.4}$$

From this statement, two very important conclusions can be drawn. The first one is that the eigenvalue of the updated problem obeys an algebraic relations involving the

updating vectors. The second one is that the eigenvector u of the original problem is still a eigenvector of the updated one.

It is natural to ask what will happen with the other eigenvalues of the system. The answer can be obtained by following a generalized version of the argument presented in (Ding and Zhou, 2007). But before doing this, two properties must be proven. The first one is written in (A.5)

$$\det(G + mn^*) = (1 + n^* G^{-1} m) \det(G) \tag{A.5}$$

A proof can be found in Ding and Zhou (2007). However, it involves the knowledge of a matrix factorization that is not motivated. Here, we present a completely alternative proof that uses only basic linear algebra facts. It starts by the application of the fact that the determinant of the product of two matrices is equal to the product of the determinants. The result can be seen in (A.6).

$$\det(G + mn^*) = \det(G) \det(\mathcal{I} + G^{-1} mn^*) \tag{A.6}$$

The determinant of a matrix is equal to the product of its eigenvalues. Therefore, the second term of this equation can be developed as in (A.7).

$$\det(\mathcal{I} + G^{-1} mn^*) = \prod_i \lambda_i(\mathcal{I} + G^{-1} mn^*) \tag{A.7}$$

It is possible to solve this eigenvalue problem explicitly. A rank-1 matrix of dimension n has $n - 1$ zero eigenvalues and only one non-zero value that is equal to the inner product of its compositing vectors. Equation (A.8) is a proof by inspection of this fact.

$$\underbrace{(ab^*)}_{rank-1}\overbrace{a}^{eigenvector} = \underbrace{(b^*a)}_{eigenvalue}\overbrace{a}^{eigenvector} \tag{A.8}$$

Another fact that must be used to figure out the product is that the eigenvalues of an arbitrary matrix are augmented by the unit if one adds the identity matrix to the original one. Therefore, all the zero eigenvalues of the rank-1 matrix are shifted to 1. Therefore, they don't influence the value of the product. The final result is given by the non-null eigenvalue of the rank-1 matrix added of one unity. This finishes the proof.

The second needed relation to show what happens to the other eigenvalues is given by (A.9).

$$\frac{1}{\lambda - \lambda_u}u = (M - \lambda_u N)^{-1}Nu \tag{A.9}$$

The proof of this is straightforward and in contained in (A.10).

$$\begin{array}{lcl} Mu & = & \lambda Nu \\ Mu - \lambda_u Nu & = & \lambda Nu - \lambda_u Nu \\ \frac{1}{\lambda-\lambda_u}u & = & (M - \lambda_u N)^{-1}Nu \end{array} \tag{A.10}$$

With these two relations, one can compute what happens to the other eigenvalues. Equations (A.11) contains this development. Notice that the first line of this development is the definition of the eigenvalues λ_u of the updated problem.

$$\begin{aligned}
\det(M - \lambda_u N + Nu(v^* - \lambda_u w^*)) &= 0 \\
(1 + (v^* - \lambda_u w^*)(M - \lambda_u N)^{-1} Nu) \det(M - \lambda_u N) &= 0 \\
\left(1 + \tfrac{v^* - \lambda_u w^*}{\lambda - \lambda_u} u\right) \det(M - \lambda_u N) &= 0 \\
\left(\lambda_u - \tfrac{\lambda + v^* u}{1 + w^* u}\right)(\lambda_{u2} - \lambda_2)(\lambda_{u3} - \lambda_3)(\lambda_{u4} - \lambda_4)\ldots &= 0
\end{aligned} \tag{A.11}$$

The final conclusion is that the updated eigenvalue in (A.4) are the same eigenvalues of the original problem with the exception of the eigenvalue associated to the eigenvector u. Therefore, this update allows the change of only one selected eigenvalue of the problem without perturbing the rest. The eigenvectors associated with the other eigenvalues are, in general, different.

A.2 Rank-N Update of the Generalized Eigenvalue Problem

This section will generalizes to a rank-N update of the generalized eigenvalue problem in (A.1). The new problem can be written simultaneously to many eigenvalues and eigenvectors as in (A.12). In this equations, D is a diagonal matrix that contains the eigenvalues. And U is a matrix whose columns contains some of the eigenvectors of the system.

$$MU = NUD \tag{A.12}$$

In the same way that was done before, the updated matrices are now given by (A.13). The main difference is that instead of having vectors to form a rank-1 update, now there are matrices of appropriated size to form rank-N updates.

$$\begin{cases} M_u = M + NUV^* \\ N_u = N + NUW^* \end{cases} \tag{A.13}$$

Using this definition, one can compute the new eigenvalues associated with the eigenvectors in the columns of U. This development is shown in (A.14). The terms in boldface were added to the equations in a way that the equality is preserved.

$$\begin{aligned} MU &= NUD \\ MU + \boldsymbol{NUV^*U} &= NUD + \boldsymbol{NUV^*U} \\ M_uU &= NU(D + V^*U) \\ M_uU &= NU\boldsymbol{(\mathcal{I} + W^*U)(\mathcal{I} + W^*U)^{-1}}(D + V^*U) \\ M_uU &= N_uU(\mathcal{I} + W^*U)^{-1}(D + V^*U) \end{aligned} \tag{A.14}$$

It is tempting to take the last line of (A.14) and use the block $(\mathcal{I} + \mathcal{W}^*U)^{-1}(D + \mathcal{V}^*U)$ as being the matrix that contains the eigenvalues of the problem. However, this matrix is not diagonal. To further develop this expression, one may use the diagonalization of this block. This is shown in (A.15).

$$(\mathcal{I} + \mathcal{W}^*U)^{-1}(D + \mathcal{V}^*U) = TD_uT^{-1} \tag{A.15}$$

Substituting (A.15) into the last line of (A.14) allows the identification of the eigenvalues and eigenvectors of this updated problem. These terms can be seen in (A.16). In this equation, the eigenvalues of the problem are the diagonal entries of D_u.

$$M_u \underbrace{UT}_{\substack{eigenvectors}} = N_u \underbrace{UT}_{\substack{eigenvectors}} \overbrace{D_u}^{eigenvalues} \tag{A.16}$$

There is an immediate questions that presents itself : is it useful to solve an eigenvalue problem in order to obtain the results of another eigenvalue problem ? This apparent contradiction is easily solved by looking at the size of the problem. In the application of interest, the original problem may have thousands of variables while the problem in (A.15) has very small size. For the application proposed in this thesis its of size two. Therefore, these small problems can normally be solved analytically.

A.3 Real Rank-2 Updates

One of the main applications of this update is moving conjugate pairs of complex eigenvalues without producing complex updated matrices. This can be achieved by a rank-2 update. In this subsection, we show that for some special configuration of the matrices U, V and W the updated matrices M_u and N_u are real.

As rank-2 updates are being considered, the matrices U, V and W have two columns. It is known that for real matrices, pairs of complex conjugate eigenvalues are associated to a pair of complex conjugate eigenvectors. Then, the matrix U has the form seen in (A.17).

$$U = \left[\begin{array}{c|c} & \\ u & u^{\dagger} \\ & \end{array}\right] \tag{A.17}$$

Taking the complex conjugate of this matrix is equivalent to inversing the order of its columns. This new matrix is then related to the original one by a permutation matrix P as shown in (A.18). One immediate property of P is that it is orthogonal.

$$U^{\dagger} = \left[\begin{array}{c|c} & \\ u^{\dagger} & u \\ & \end{array}\right] = U \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = UP \tag{A.18}$$

If one choses the matrices V and W as having the same structure of U, it is possible to compute the complex conjugate of the rank-2 update matrix and obtain that it is equal to itself, meaning that this update is real. This development is made in (A.19).

$$(NUV^*)^\dagger = N^\dagger U^\dagger V^T = NUPP^T V^* = NUV^* \tag{A.19}$$

Therefore, if the matrix of the original eigenvalue problem is real, this kind of update will not add any complex numbers to it. This behavior is exploited in the thesis to produce stable models that remain in the real field.

Appendix B

Subspace produced by the modified Arnoldi Iteration

The main objective of this appendix is to show that the subspace computed by the classical Arnoldi Iteration and the alternative version proposed in this thesis are same for the case of some special matrices commonly used to perform model order reduction.

This appendix is divided in three main sections. The first one presents the two subspaces whose equivalence we are trying to show and explains in details the approach that will be taken. The second section has some math background that will be useful for the reader to understand the following development in a self-contained manner. Finally, the third subsection will perform the proof itself.

B.1 Statement of the problem

The Krylov subspace corresponds to the subspace spanned by the vectors in (B.1). The matrix K is called the generating matrix and the vector v is the generating vector. At this point these two entities are arbitrary.

$$\mathcal{K}\{K,v\} = \text{span}\{v,Kv,K^2v,K^3v,\ldots\} \tag{B.1}$$

In the framework of model order reduction by moment matching, one has to compute a matrix V whose span is the Krylov subspace given by (B.2).

$$\text{span}\{V\} = \mathcal{K}\{(A - s_0E)^{-1}E,(A - s_0E)^{-1}B\} \tag{B.2}$$

In practice, this series will be stopped at a given position and only a finite number of moments are matched. For simplicity, we analyze only the two first vectors of this subspace. This will not compromise the generality of the results because, as we will demonstrate shortly, it is possible to apply the argument many times for each new added pair of vectors. Also, as the orthogonalization process do not change the span of the matrices, we will not include this step into the analysis. The first and second vectors of (B.2) were arranged into a matrix V shown in (B.3).

$$V = \left[\begin{array}{c|c} v & Kv \end{array}\right] \tag{B.3}$$

This matrix is complex in the general case because the expansion point s_0 is normally chosen to be purely imaginary. A standard way of producing a real matrix that spans the column space of V is to separate its real and imaginary parts and build an augmented matrix with both. The new matrix is given by (B.4).

$$V_{real} = \left[\begin{array}{c|c|c|c} \Re\{v\} & \Im\{v\} & \Re\{Kv\} & \Im\{Kv\} \end{array}\right] \tag{B.4}$$

On the other hand, using the modified Arnoldi Iteration described in the thesis, the subspace of the new V matrix is equivalent to the one of the matrix V_{new} in (B.5).

$$V_{new} = \left[\begin{array}{c|c|c|c} \Re\{v\} & \Im\{v\} & \Re\{K\Im\{v\}\} & \Im\{K\Im\{v\}\} \end{array}\right] \tag{B.5}$$

At this point the main objective of this appendix can be stated. We are going to show that the span of (B.5) and (B.4) are the same for the special matrices in (B.2). This

means that all the columns of V_{real} can be reconstructed by linear combinations of the columns of V_{new}. In a linear algebra approach, it means that there exists a non-singular matrix T such that the relation in (B.6) holds.

$$V_{real} = V_{new}T \tag{B.6}$$

Therefore, the main objective is the demonstrations that such a T exists.

B.1.1 Mathematical background

In this section, the expansion of real and imaginary parts of the generating matrices are explained in detail. The main relation that will be used in our developments is given by (B.7).

$$(\mathcal{I} - M)^{-1} = \sum_{i=0}^{\infty} M^i \tag{B.7}$$

It is possible to recognize this expression as the generalization of the sum of the geometric series for matrices. Equations (B.8) contains the proof of this relation.

$$\begin{array}{lcll} S & = & \mathcal{I} + M + M^2 + M^3 + \ldots & \textbf{(a)} \\ SM & = & M + M^2 + M^3 + M^4 \ldots & \textbf{(b)} \\ S(\mathcal{I} - M) & = & \mathcal{I} & \textbf{(a) - (b)} \\ S & = & (\mathcal{I} - M)^{-1} & \end{array} \tag{B.8}$$

Let the matrix M be purely imaginary. Then, it is reasonable to imagine that the expression in (B.7) will be complex. Therefore, it is possible to analyze separately its real and imaginary parts. To deduce this expressions, we start by looking at (B.9). In

this equations, (B.7) has been directly applied ignoring the fact that the matrix M is now complex.

$$(\mathcal{I} - j\omega K)^{-1} = \sum_{i=0}^{\infty} (j\omega K)^i \tag{B.9}$$

This sum can be expanded and the even and odd terms will give rise to its real and imaginary parts. The complete development is done in (B.10).

$$\begin{aligned}
\textstyle\sum_{i=0}^{\infty}(j\omega K)^i &= (j\omega K)^0 + (j\omega K)^1 + (j\omega K)^2 + (j\omega K)^3 + (j\omega K)^4 + (j\omega K)^5 \dots \\
&= \left[(j\omega K)^0 + (j\omega K)^2 + (j\omega K)^4 + \dots\right] + \left[(j\omega K)^1 + (j\omega K)^3 + (j\omega K)^5 + \dots\right] \\
&= \left[[\mathcal{I}]^0 + [(j\omega K)^2]^1 + [(j\omega K)^2]^2 + \dots\right] + j\omega K\left[\mathcal{I} + [(j\omega K)^2]^1 + [(j\omega K)^2]^2 + \dots\right] \\
&= (\mathcal{I} + \omega^2 K^2)^{-1} + j\omega K(\mathcal{I} + \omega^2 K^2)^{-1}
\end{aligned} \tag{B.10}$$

The same idea can be used to separate the real and imaginary parts of the inverse of the matrix pencil $A - s_0 E$. We start by factoring out A from this expression in order to allow direct application of (B.7). The result is shown in (B.11). In this equations, we are assuming that the matrix A is non-singular.

$$(A - s_0 E)^{-1} = (\mathcal{I} - s_0 A^{-1} E)^{-1} A^{-1} = [U + j\omega L U] A^{-1} \tag{B.11}$$

To simplify this last expression two new matrices were defined, the first one is U in (B.12). The second one is L, defined in (B.13).

$$U = (\mathcal{I} + \omega^2 L^2)^{-1} \tag{B.12}$$

$$L = A^{-1}E \tag{B.13}$$

From the development (B.10), it should be clear that matrices U and L commute. In other words, $UL = LU$.

Making use of these recently defined matrices, the generating vector for the moment matching technique is then given by (B.14).

$$(A - s_0E)^{-1}b = Ub_0 + j\omega LUb_0 \tag{B.14}$$

Another vector b_0 has been defined to lighten the notation. Its definition can be found in (B.15).

$$b_0 = A^{-1}b \tag{B.15}$$

In the same way, the generating matrix for moment matching is given by (B.16)

$$(A - s_0E)^{-1}E = UL + j\omega L^2U \tag{B.16}$$

These way of writing the generating matrix and vector are very useful because it explicits the structure present in these entities. In the next section this will become very clear.

B.2 Proof of Equivalence of Subspaces

We start by computing the product Kv in terms of the matrices defined in the last section. This product is written in (B.17).

$$(UL + j\omega L^2U)(Ub_0 + j\omega LUb_0) = (U^2Lb_0 - \omega^2U^2L^3b_0) + j\omega(U^2L^2b_0 + U^2L^2b_0) \tag{B.17}$$

Using this new result, the matrix (B.4) can be explicitly written in (B.18).

$$V_{real} = \left[\begin{array}{c|c|c|c} Ub_0 & \omega LUb_0 & (U^2L - \omega^2U^2L^3)b_0 & 2\omega U^2L^2b_0 \end{array}\right] \tag{B.18}$$

For the case of the proposed Arnoldi iteration, the matrix K acts only in the imaginary part of the vector. In terms of the matrices defined in the last section, this product results in (B.19).

$$(UL + j\omega L^2U)(\omega LUb_0) = \omega U^2L^2b_0 + j\omega^2U^2L^3b_0 \tag{B.19}$$

Using this product, the matrix V_{new} can be built using the matrices defined in the last section. This result is shown in (B.20).

$$V_{new} = \left[\begin{array}{c|c|c|c} Ub_0 & \omega LUb_0 & \omega U^2L^2b_0 & \omega^2U^2L^3b_0 \end{array}\right] \tag{B.20}$$

At this point it is possible to see that the first and second columns of both matrices are the same which, evidently, span the same two dimensional subspace. Fixing the attention at the third and forth columns, one can verify that the forth column in V_{real} and the third column in V_{new} are related by a constant factor. Therefore, these columns span the same one dimensional subspace

It is hard to find a way of reconstructing the third column of V_{real} by a linear combination of the columns of V_{new} by simple inspection. However, this is possible to be done by the development of the imaginary part of v, or the second columns of (B.20). The result of this development is contained in (B.21). In this equation, only the definition of U has been used.

$$\omega LUb_0 = \omega LU^2(\mathcal{I} + \omega^2 L^2)b_0 = \omega LU^2 b_0 + \omega^3 L^3 U^2 b_0 \tag{B.21}$$

This allows the consideration of the same matrix V_{new} but in a different form contained in (B.22).

$$V_{new} = \left[\begin{array}{c|c|c|c} Ub_0 & \omega LU^2 b_0 + \omega^3 L^3 U^2 b_0 & \omega U^2 L^2 b_0 & \omega^2 U^2 L^3 b_0 \end{array}\right] \tag{B.22}$$

This new way of expressing the matrix reveals a way of constructing the third columns of V_{real} by a linear combination of the second and fourth columns of V_{new}. A mathematical way of writing this is the existence of constants α and β that verifies the equality in (B.23).

$$\alpha(\omega LU^2 b_0 + \omega^3 L^3 U^2 b_0) + \beta\omega^2 U^2 L^3 b_0 = (U^2 L - \omega^2 U^2 L^3)b_0 \tag{B.23}$$

It is clear that choosing $\alpha = \omega^{-1}$ and $\beta = -2$ verifies the equality, finally proving that is possible to reconstruct all the columns of V_{real} by linear combinations of the columns

of V_{new}. In other words, both matrices span the same subspace. The matrix T that represents this relations is given by (B.24). As one can see, its determinant is always 2 what proves that it is also non-singular.

$$V_{real} = V_{new} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & \frac{1}{\omega} & 0 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & -2 & 0 \end{bmatrix}}_{T} \tag{B.24}$$

The same idea used to prove the case of only four vectors of the subspace can be extended to a subspace of any dimension. In order to stat this argument, notice that the first columns of (B.18) contains only the first powers of U and that the third and fourth columns have only terms in U^2. This pattern can be extended. The next two columns will contain pairs in U^3 and so on.

It is possible to reconstruct these columns of V_{real} by the successive applications of (B.12) and the obvious fact that $UU^{-1} = \mathcal{I}$. This can be used to change the exponents of U and L in order to produce the desired linear combinations. As an example, we present the matrix T for the case of two subspaces up to 14 vectors, contained in (B.25). Notice that the sub-matrices can be used to show the equivalence of subspaces of size 2,4,6,...,14.

$$T_{14} = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & \frac{1}{\omega} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 2 & \frac{1}{\omega} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -2 & 0 & 0 & \frac{3}{\omega} & \frac{1}{w^2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -4 & 0 & 0 & \frac{4}{\omega} & \frac{1}{\omega^2} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -4 & \frac{-8}{\omega} & 0 & 0 & \frac{5}{\omega^2} & \frac{1}{\omega^3} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -8 & \frac{-12}{\omega} & 0 & 0 & \frac{6}{\omega^2} & \frac{1}{\omega^3} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 8 & 0 & 0 & \frac{-20}{\omega} & \frac{-18}{\omega^2} & 0 & 0 & \frac{7}{\omega^3} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 16 & 0 & 0 & \frac{-32}{\omega} & \frac{-24}{\omega^2} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 16 & \frac{48}{\omega} & 0 & 0 & \frac{-56}{\omega^2} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 32 & \frac{80}{\omega} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -32 & 0 & 0 & \frac{112}{\omega} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -64 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -64
\end{bmatrix} \tag{B.25}$$

Appendix C

Identities Involving Vectors

C.1 Introduction

In this appendix some very useful vector identities are derived. They concern mainly of simple rules that one can apply to vector expression in order to obtain their derivatives. This is of great help because there will be no need to convert matrix expression into indicial form, perform the derivation, and transform back to matrix notation.

C.2 Derivatives

Following what has been said in the section 1.5 about notation, the symbol ∇_x is used to represent the derivative of a quantity in respect to each one of the entries of $x \in \mathbb{R}^n$. It can then be understood as being an vectorial operator as shown in (C.1).

$$\nabla_x = \begin{bmatrix} \frac{\partial}{\partial x_1} \\ \frac{\partial}{\partial x_2} \\ \vdots \\ \frac{\partial}{\partial x_n} \end{bmatrix} \tag{C.1}$$

If this symbol is applied to the transpose of x, it produces the identity matrix ($\nabla_x x^T = \mathcal{I}$). One can obtain this by using the identity (C.2).

$$\frac{\partial x_i}{\partial x_j} = \delta_j^i = \begin{cases} 1 & \text{if} \quad i = j \\ 0 & \text{if} \quad i \neq j \end{cases} \tag{C.2}$$

There are two rules for the behavior of ∇_x that will facilitate immensely the process of deriving complex expressions. It is natural to wonder what happens if some constant matrix $A \in \mathbb{R}^{n \times m}$ multiplies the vector x. The resulting process is shown in (C.3).

$$\nabla_x (x^T A) = A \tag{C.3}$$

Intuition can be used to provide understating of this equations. As the matrix A is constant, is does not depend of x and can be factored out of the expression. Therefore, the operator ∇_x only acts on x^T witch results in the identity, giving the final result A. Of course this can be made precise, as shown in (C.4).

$$\begin{aligned} \left[\nabla_x \left(x^T A\right)\right]_{ij} &= \tfrac{\partial}{\partial x_i} \left(\sum_k x_k A_{kj} \right) = \sum_k \frac{\partial x_k}{\partial x_i} A_{kj} \\ &= \sum_k \delta_i^k A_{kj} = A_{ij} \end{aligned} \tag{C.4}$$

The second rule that we shall derive is an analog to the product rule in one variable calculus. If we consider u and v columns vectors of appropriated size, the behavior of ∇_x when applied to the scalar product $u \cdot v$ is given by (C.5).

$$\nabla_x (u^T v) = (\nabla_x u^T) v + (\nabla_x v^T) u^T \tag{C.5}$$

The intuition behind this expression is that it behaves just as the normal product rule but a transposition in the second term is needed to assure the dimensional consistency of the matrix multiplication (the product of two vectors is just a special case). As we did before, the above relation can be rigorously proven as shown in (C.6).

$$\begin{aligned}
\left[\nabla_x(u^T v)\right]_i &= \tfrac{\partial}{\partial x_i}\left(\sum_k u_k v_k\right) \\
&= \sum_k \frac{\partial u_k}{\partial x_i} v_k + \sum_k u_k \frac{\partial v_k}{\partial x_i} \\
&= \sum_k \frac{\partial u_k}{\partial x_i} v_k + \sum_k \frac{\partial v_k}{\partial x_i} u_k \\
&= \left[(\nabla_x u^T)v + (\nabla_x v^T)u\right]_i
\end{aligned} \tag{C.6}$$

Combining only these two relations, many different complicated expressions can be derived. We present some examples in (C.7) and in (C.8).

$$\nabla_x(u^T A v) = (\nabla_x u^T)Av + (\nabla_x v^T)Au \tag{C.7}$$

$$\nabla_x\left(\frac{1}{2}x^T Ax - b^T x + c\right) = \left(\frac{A + A^T}{2}\right)x - b \tag{C.8}$$

C.2.1 Validity of the covariance kernel

If a matrix A is defined by the function $k : \mathbb{R}^d \times \mathbb{R}^d \longrightarrow \mathbb{R}$ such that $A_{ij} = k(x_i, x_j)$ and k is symmetric with respect to its entries, then k is called the kernel of the matrix A. If A is positive (semi)-definite then k is a (semi)-definite positive kernel.

It is known that if the function k has the form $k = k(\|x_i - x_j\|)$ then it is a positive definite kernel if all the points x_i are distinct.

We start by showing that the linear kernel is positive definite. This can be easily shown by direct application of the definition as exposed in .

$$\begin{aligned} c^T K c &= \sum_{i,j} c_i c_j u_i^T u_j \\ &= \sum_{i,j} c_i c_j \sum_k (u_i)_k (u_j)_k \\ &= \sum_k \sum_i c_i (u_i)_k \sum_j c_j (u_j)_k \\ &= \sum_k \left(\sum_i c_i (u_i)_k \right)^2 \geq 0 \end{aligned} \tag{C.9}$$

We are going to start from this kernel and perform various transformation, each one preserving positive definiteness, until the Gaussian kernel is reached. It should be evident that multiplication by a positive scalar and the sum of any number of positive definite kernels preserves the positive definiteness. It is not so clear that this property is preserved in the product of two such functions. Therefore, we proceed to show that this is the case.

Assume two positive definite kernels denoted by the functions $a(x,y)$ and $b(x,y)$ which produces the matrices $A,B \geq 0$. We have to show that the newly formed matrix $C_{ij} = A_{ij}B_{ij}$ by product of the functions a and b is positive semidefinite. To do so, we start by writing A as its eigenvalue decomposition $A = U\Lambda U^T$. This decomposition is aways possible because the matrix A is symmetric. The development of the proof is done in .

$$\begin{aligned} c^T K c &= \sum_{i,j} c_i c_j C_{ij} \\ &= \sum_{i,j} c_i c_j A_{ij} B_{ij} \\ &= \sum_{i,j} c_i c_j B_{ij} \sum_k \lambda_k U_{ik} U_{jk} \\ &= \sum_k \lambda_k \sum_{i,j} c_i U_{ik} B_{ij} c_j U_{jk} \\ &= \sum_{i,j} \lambda_k \sum_k \eta_k^T B \eta_k \geq 0 \end{aligned} \tag{C.10}$$

We have set $\eta_k = [\ c_1 U_{1k}\ \ c_2 U_{2k}\ \ \dots\ ]^T$. An extremely similar procedure can be used to prove that the kernel $k'(x,y) = f(x)k(x,y)f(y)$, composed of a positive definite kernel $k(x,y)$ and a function $f : \mathbb{R}^n \to \mathbb{R}$ also preserves positive definiteness.

The last step before concluding that the Gaussian kernel is positive definite is to write it as three different functions by expanding its argument. This is shown in (C.11).

$$e^{\epsilon\|u-v\|^2} = e^{-\epsilon u^T u} e^{2\epsilon u^T v} e^{-\epsilon v^T v} \tag{C.11}$$

At this point, everything that is needed to perform the proof has been presented. One starts by taking the linear kernel and performing the multiplication by 2ϵ (A). As the exponential function consists of powers of the argument and multiplications by positive numbers, it preserves positive definiteness (B). Use the function composition rule selecting $\exp(-\epsilon x^T x)$ as its argument (C). Contract the product to obtain the final function (D). This chain of arguments is illustrated below.

$$u^T v \xrightarrow{A} 2\epsilon u^T v \xrightarrow{B} e^{2\epsilon u^T v} \xrightarrow{C} e^{-\epsilon u^T u} e^{2\epsilon u^T v} e^{-\epsilon v^T v} \xrightarrow{D} e^{-\epsilon\|u-v\|^2}$$

Appendix D

Electric Circuit Equations

Introduction

The objective of this section is the deduction of the circuit equations for a specific configuration of components as shown in D.1. We assume that all the inductors are in series with a resistor. Therefore, we are dealing with only four circuit "elements". Note that D.1 contains the sign convention that we use.

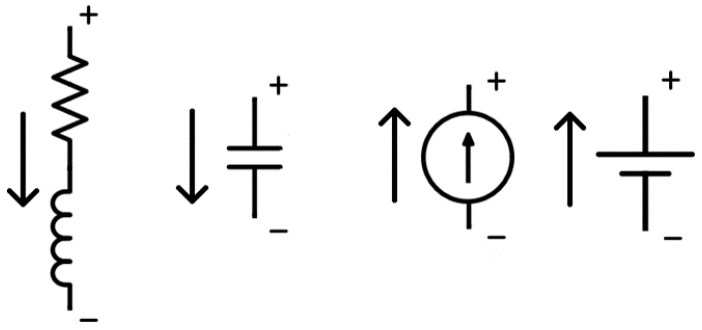


Figure D.1: Circuit elements and sign convention

A circuit C is an oriented graph composed of a set N of n nodes and a set E of e edges. The edges are ordered pairs (N_i, N_j) of nodes. The nodes N_i and N_j are called tail and head of the edge, respectively. The connection between these elements can be stored in an incidence matrix $I \in \mathbb{R}^{e \times n}$ that is defined in (D.1).

$$I_{ij} = \begin{cases} 1 & \text{if} \quad \text{head}(E_i) = N_j \\ -1 & \text{if} \quad \text{tail}(E_i) = N_j \\ 0 & \quad \text{otherwise} \end{cases} \tag{D.1}$$

As we are dealing with conservative electric fields, adding a constant to the potential at all nodes do not change the current that flows through the edges. Therefore, we must

eliminate this degree of freedom if we want the solution of our set of equations to be unique. This can be done by choosing one node of the system and fixing its potential to zero. In the incidence matrix (D.1), this is equivalent to deleting the column that corresponds to this node. Henceforth, we assume that I has been through this process.

We can assume without loss of generality that the edges are ordered according to their type of component. Then, one can partition the incidence matrix into smaller ones containing information only of their correspondent component. This partitioning is shown in (D.2). The indexes should be self explanatory.

$$I = [\ I_{rl}\ \ I_c\ \ I_i\ I_v\ ] \tag{D.2}$$

Using one of these matrices, one can write the relation that exists between the potential difference in points where voltage sources are attached. This relation is shown in (D.3). In this equations, v_s represents the voltage value imposed at the sources.

$$I_v v = v_s \tag{D.3}$$

The Kirchhoff current law can be written in matrix form as in (D.4).

$$I^T i = I_{rl}^T i_{rl} + I_c^T i_c - I_i^T i_i - I_v^T i_v = 0 \tag{D.4}$$

This equations can be further developed by substituting the current of the capacitor by its standard relation. Doing this results in (D.5).

$$I_{rl}^T i_{rl} + I_c^T C \frac{dv_c}{dt} - I_v^T i_v = I_i i_i \tag{D.5}$$

The voltages at the capacitor can be expressed as a function of the vector of voltages of the system ($v_c = I_c v$). Performing this last substitution one can arrive at the first equations that describes de circuit. This result is shown in (D.6).

$$I_{rl}^T i_{rl} + I_c^T C I_c \frac{dv}{dt} = I_i^T i_i + I_v^T i_v \tag{D.6}$$

The second equation can be obtained directly by the circuit relations between voltage and current of the element containing the resistor and inductor, as in (D.7).

$$v_{rl} = R i_{rl} + L \frac{di_{rl}}{dt} \tag{D.7}$$

Using the same idea used to the capacitors, it is possible to obtain a relation of the voltage in these elements as a function of the voltage vector of the circuit ($v_{rl} = I_{rl} v$). Substituting this relation into (D.7), one can obtain the second equation for this kind of circuit (D.8).

$$I_{rl} v = R i_{rl} + L \frac{di_{rl}}{dt} \tag{D.8}$$

Equations (D.3), (D.6) and (D.8) can be put together and the state space representation is shown in (D.9).

$$\begin{bmatrix} I_c^T C I_c & 0 & 0 \\ 0 & L & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{dv}{dt} \\ \frac{di_{rl}}{dt} \\ \frac{di_v}{dt} \end{bmatrix} = \begin{bmatrix} 0 & -I_{rl}^T & I_v^T \\ I_{rl} & -R & 0 \\ -I_v & 0 & 0 \end{bmatrix} \begin{bmatrix} v \\ i_{rl} \\ i_v \end{bmatrix} + \begin{bmatrix} I_i^T & 0 \\ 0 & 0 \\ 0 & \mathcal{I} \end{bmatrix} \begin{bmatrix} i_i \\ v_s \end{bmatrix} \tag{D.9}$$

Note that we have not explicitly written the output equations due to the multitude of variables of interest that one may probe. Each one of them would produce different output matrices.

At the specific case of impedance probing, there is no voltage source but only one current source at the probing point. The output matrix uses the current incidence matrix to obtain the voltage. Once that the matrices C, L and R are positive definite, one can directly conclude that this state space can be put in the form of (D.10) and $E \succeq 0$ and $A \preceq 0$.

$$\begin{cases} E\frac{dx}{dt} &= Ax + Bu \\ y &= B^T u \end{cases} \tag{D.10}$$

Appendix E

Kriging

E.1 Introduction

In chapter 5 we have presented the theory behind Kriging. However, many intermediate steps were left behind and only final results were presented. This appendix aims at filling these gaps with the missing steps. Although they are not important for direct implementation of the methods, they are needed for understanding it in a deeper level.

E.2 No Bias Condition

Imposing the value zero for the expected mean value of the error and applying the definition of the matrix F, one obtains the no bias condition in (E.1).

$$\begin{aligned} E[\Delta] &= E\left[\beta^T Y - \hat{Y}(x_0)\right] \\ 0 &= \beta^T F\alpha - f(x_0)^T\alpha \\ F^T\beta &= f(x_0) \end{aligned} \tag{E.1}$$

In the second line of this process, we have used the fact that there should be no bias regardless of the value estimated for α.

E.3 The variance identity

There is a very useful identity for the variance involving inner products of vectors, given by (E.2).

$$\text{var}\{u^T\alpha + v^T\omega\} = u^T\text{var}\{\alpha\}u + v^T\text{var}\{\omega\}v + u^T\text{cov}\{\alpha,\omega\}v + v^T\text{cov}\{\omega,\alpha\}u \tag{E.2}$$

The variables u and v are vectors in $\mathbb{R}^{na}$ and $\mathbb{R}^{nb}$, respectively and α and ω are stochastic vectors in $\mathbb{R}^{na}$ and $\mathbb{R}^{nb}$ respectively. The deduction of this identity, although tedious, is very straight forward and can be obtained directly by the application of the definition of variance and covariance, as shown in .

$$\begin{aligned}
\text{var}\{u^T\alpha + v^T\omega\} &= E\left[(u^T\alpha + v^T\omega - E[u^T\alpha + v^T\omega])^2\right] \\
&= E[(u^T(\alpha - E[\alpha]) + v^T(\omega - E[\omega]))^2] \\
&= E[(\mathcal{A} + \mathcal{B})^2] \\
&= E[\mathcal{A}^2 + \mathcal{B}^2 + 2\mathcal{A}\mathcal{B}] \\
&= u^T\text{var}\{\alpha\}u + v^T\text{var}\{\omega\}v + 2u^T\text{cov}\{\alpha,\omega\}v
\end{aligned} \tag{E.3}$$

Note that as $\mathcal{A}$ is a scalar, its square can be written as $\mathcal{A}^2 = \mathcal{A}\mathcal{A}^T$. A similar argument is valid for $\mathcal{B}$.

E.4 Variance of the Error

Applying (E.2) to (5.12) results in the variance for the error. This development can be found in (E.4).

$$\begin{aligned}
\text{var}\{\Delta\} &= \text{var}\{\beta^T Y - \hat{Y}(x_0)\} \\
&= \text{var}\{\hat{Y}(x_0)\} + \beta^T\text{var}\{Y\}\beta - \text{cov}\{\hat{Y}(x_0),Y\}\beta - \beta^T\text{cov}\{Y,\hat{Y}(x_0)\} \\
&= \text{var}\{\hat{Y}(x_0)\} + \beta^T\text{cov}\{Y\}\beta - 2\beta^T\text{cov}\{Y,\hat{Y}(x_0)\} \\
&= \sigma^2(1 + \beta^T R\beta - 2\beta^T r)
\end{aligned} \tag{E.4}$$

E.5 Solving the restricted problem

To obtain the values for the coefficients of β, the variance of the error should be minimized subjected to the no bias condition. We start by writing the Lagrangian function for the problem. We have omitted continuous terms and positive multiplicative constants that do not alter the solution. This function is shown in (E.5). Note that the Lagrange multipliers are the entries of the vector 2λ. We have chosen this multiplicative constant to simplify the computation.

$$\mathcal{L} = \beta^T R\beta - 2\beta^T r + 2\lambda^T(F^T\beta - f(x_0)) \tag{E.5}$$

Taking the gradient with respect to λ recovers the no bias condition and is very straight forward. The gradient with respect to β is only a little more complicated and the process is contained in in (E.6).

$$\begin{array}{rcl} \nabla_\beta \mathcal{L} & = & \nabla_\beta \left(\beta^T R\beta - 2\beta^T r + 2\lambda^T(F^T\beta - f(x_0))\right) \\ 0 & = & 2R\beta - 2r + 2F\lambda \end{array} \tag{E.6}$$

These two equations give rise to the linear system shown in (5.16).

E.6 Maximum Likelihood Estimation

During the presentation of the method, various parameters had to be estimated from the data. They were obtained by means of the Maximum Likelihood Estimations (MLE). We do not aim at presenting a exhaustive or profound discussion about this topic. We only present the subject with enough detail so that the reader can understand the results obtained in chapter 5.

In the statistical community, the most famous and widely used probability density function is the Gaussian, whose equations is given by (E.7).

$$\mathcal{G}(x) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right) \tag{E.7}$$

This equations, however, only represents the Probability Distribution Function (pdf) in one dimension. If we assume the existence of many independent variables denoted by x_i with zero mean and standard deviations σ_i, the joint PDF is then given by (E.8).

$$\prod_{i=1}^{m}\mathcal{G}(x_i) = (2\pi)^{-\frac{n}{2}}\left(\prod_{i}^{m}\sigma_i^{-1}\right)\exp\left(-\frac{1}{2}\sum_{i}^{m}\frac{x_i^2}{\sigma_i^2}\right) \tag{E.8}$$

This function cab be written in a more concise manner by the use of matrix notation. Define $V \in \mathbb{R}^{m\times m}$ as in (E.9).

$$V_{ij} = \begin{cases} \sigma_i^2 & \text{if} \quad i = j \\ 0 & \text{if} \quad i \neq j \end{cases} \tag{E.9}$$

Then, (E.8) can be written as (E.10). Henceforth, we regard x as being a vector containing the individual entries x_i.

$$\prod_{i=1}^{m}\mathcal{G}(x_i) = (2\pi)^{-\frac{n}{2}}\sqrt{\det(V)}\exp\left(-\frac{1}{2}x^T V^{-1}x\right) \tag{E.10}$$

Up to this point, the variables have been assumed to be independent. If one defines a rotation and a new set of coordinates given by $x = Q\tilde{x}$ with $Q^T = Q^{-1}$, then (E.10) keep its form with $\tilde{x}$ and $\widetilde{V}$ given by $\widetilde{V} = Q^T V Q$. Figure E.1 has a depiction of the original pdf with independent variables and the rotated one with dependent variables.

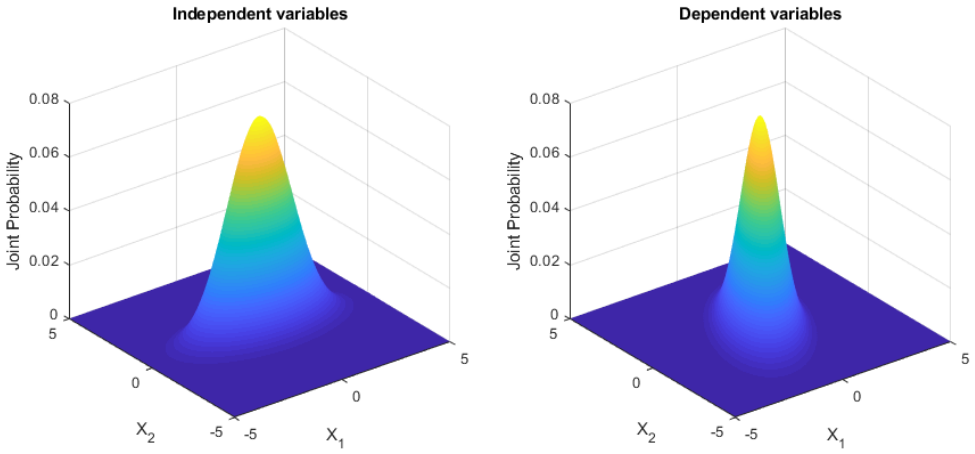


Figure E.1: Example of two Gaussians, one with independent variables and one with dependent ones.

Having established the formulas for the multivariate Gaussian we are able to present the idea behind Maximum Likelihood Estimation. We have supposed that the values obtained by evaluating the expensive functions were samples from a random variable constituted by a deterministic portion and a stochastic process with zero mean and a given covariance matrix. The joint probability density function of the process is given by (E.11).

$$\mathcal{G}(x_1, x_2, \ldots, x_m) = (2\pi)^{-\frac{n}{2}} \det(V)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(y - F\alpha)^T V^{-1}(y - F\alpha)\right) \tag{E.11}$$

The MLE consists in finding the values for the parameters of this distribution that maximizes the value of the joint probability function. Using the jargon, one must maximize the likelihood of the data. As the logarithm function is monotonically increasing, it can be applied to (E.11) to facilitate the derivation process. This new function is called log-likelihood function and is written out in (E.12).

$$\log \mathcal{G}(x_1, x_2, \ldots, x_m) = -\frac{n}{2}\log 2\pi - \frac{1}{2}\log(|V|) - \frac{1}{2}(y - F\alpha)^T V^{-1}(y - F\alpha) \tag{E.12}$$

Taking the gradient of this function with respect to α and making it equals to zero, wields its MLE shown in (E.13).

$$\hat{\alpha} = (F^T V^{-1} F)^{-1} F^T V^{-1} y \tag{E.13}$$

To ensure that this is a maximum, one has to compute the Hessian, which results in $-F^T V^{-1} F/\sigma^2$. Knowing that V is positive semidefinite, this expression is negative semidefinite, assuring that the obtained value for $\hat{\alpha}$ is indeed a maximum.

The same process can be repeated but taking the partial derivative with respecting to σ to obtain $\hat{\sigma}$. The result is shown in (E.14).

$$\hat{\sigma}^2 = \frac{1}{n}(y - F\alpha)^T R^{-1}(y - F\alpha) \tag{E.14}$$

Note that we have used the definition $V = \sigma^2 R$.

E.7 Explicit solution for the Kriging linear system

One could solve the linear system (5.16) for each new value of x_0. This is, however, not the best practice, since the matrix R is not dependent of the estimation. One best way of solving this system is to use its explicit block inverse.

This inverse can be obtained by direct Gaussian elimination. One starts with an augmented matrix as shown in (E.15). Performing Gaussian elimination in blocks, one must reduce the left hand side of this expression into the block identity matrix. What is left in the right hand side is the block inverse of the left hand side.

$$\left[\begin{array}{cc|cc} R & F & \mathcal{I} & 0 \\ F^T & 0 & 0 & \mathcal{I} \end{array}\right] \tag{E.15}$$

Performing these operations lead to the identity (E.16).

$$\begin{bmatrix} R & F \\ F^T & 0 \end{bmatrix}^{-1} = \begin{bmatrix} R^{-1} - R^{-1}B(F^TR^{-1}F)^{-1}B^TR^{-1} & R^{-1}F(F^TR^{-1}F)^{-1} \\ (F^TR^{-1}F)^{-1}F^TR^{-1} & -(F^TR^{-1}F)^{-1} \end{bmatrix} \tag{E.16}$$

Note that as R is positive definite, it can be decomposed by means of a Cholesky factorization. If for any numerical reasons it becomes positive semidefinite, then one can add a very small value (e.g. 1×10^{-10}) to its diagonal and force all the eigenvalues to be positive.

Defining (E.17) allows the simplification of the following expressions.

$$\Psi = (F^TR^{-1}F)^{-1}F^TR^{-1} \tag{E.17}$$

Finally, using (E.16), the values for β and λ can be explicitly written as in and .

$$\beta = R^{-1}(\mathcal{I} - F\Psi)r + \Psi^T f \tag{E.18}$$

$$\lambda = \Psi r - (F^TR^{-1}F)^{-1}f \tag{E.19}$$

Using the maximum likelihood estimator for α allows the writing of the predicted value in a simple manner (E.20).

$$\beta^T y = f^T\hat{\alpha} + r^TR^{-1}(y - F\hat{\alpha}) \tag{E.20}$$

Bibliography

Absil, P.-A., Mahony, R., and Sepulchre, R. (2009). *Optimization algorithms on matrix manifolds*. Princeton University Press.

Amsallem, D. (2010). *Interpolation on manifolds of CFD-based fluid and finite element-based structural reduced-order models for on-line aeroelastic predictions*. PhD thesis, Stanford University.

Amsallem, D. and Farhat, C. (2008). Interpolation method for adapting reduced-order models and application to aeroelasticity. *AIAA journal*, 46(7):1803.

Amsallem, D. and Farhat, C. (2012). Stabilization of projection-based reduced-order models. *International Journal for Numerical Methods in Engineering*, 91(4):358–377.

Antoulas, A. C. (2005). *Approximation of large-scale dynamical systems*, volume 6. Siam.

AO Leite, M., Delinchant, B., Guichon, J.-M., and A Vasconcelos, J. (2016). Simplex-based adaptive parametric model order reduction for applications in optimization. *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields*.

Arnoldi, W. E. (1951). The principle of minimized iterations in the solution of the matrix eigenvalue problem. *Quarterly of applied mathematics*, 9(1):17–29.

Ayed, R. B., Gong, J., Brisset, S., Gillon, F., and Brochet, P. (2012). Three-level output space mapping strategy for electromagnetic design optimization. *IEEE Transactions on Magnetics*, 48(2):671–674.

Bakr, M. H., Bandler, J. W., Madsen, K., and Søndergaard, J. (2000). Review of the space mapping approach to engineering optimization and modeling. *Optimization and Engineering*, 1(3):241–276.

Balanis, C. A. (1999). *Advanced engineering electromagnetics*. John Wiley & Sons.

Ball, K. (1992). Eigenvalues of euclidean distance matrices. *Journal of Approximation Theory*, 68(1):74–82.

Bandler, J., Biernacki, R., Chen, S., Grobelny, P., and Hemmers, R. (1994a). Exploitation of coarse grid for electromagnetic optimization. In *Microwave Symposium Digest, 1994., IEEE MTT-S International*, pages 381–384. IEEE.

Bandler, J. W., Biernacki, R. M., Chen, S. H., Grobelny, P. A., and Hemmers, R. H. (1994b). Space mapping technique for electromagnetic optimization. *IEEE Transactions on Microwave Theory and Techniques*, 42(12):2536–2544.

Bartels, R. H. and Stewart, G. (1972). Solution of the matrix equation ax+ xb= c [f4]. *Communications of the ACM*, 15(9):820–826.

Baudoui, V. (2012). *Optimisation robuste multiobjectifs par modèles de substitution*. PhD thesis, Toulouse, ISAE.

Benner, P., Gugercin, S., and Willcox, K. (2015). A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM review*, 57(4):483–531.

Berkooz, G., Holmes, P., and Lumley, J. L. (1993). The proper orthogonal decomposition in the analysis of turbulent flows. *Annual review of fluid mechanics*, 25(1):539–575.

Bond, B. N. and Daniel, L. (2008a). Guaranteed stable projection-based model reduction for indefinite and unstable linear systems. In *Proceedings of the 2008 IEEE/ACM International Conference on Computer-Aided Design*, pages 728–735. IEEE Press.

Bond, B. N. and Daniel, L. (2008b). Guaranteed stable projection-based model reduction for indefinite and unstable linear systems. In *Proceedings of the 2008 IEEE/ACM International Conference on Computer-Aided Design*, pages 728–735. IEEE Press.

Bondeson, A., Rylander, T., and Ingelström, P. (2005). *Computational electromagnetics*, volume 51. Springer Science & Business Media.

Bouhamidi, A., Heyouni, M., and Jbilou, K. (2011). Block arnoldi-based methods for large scale discrete-time algebraic riccati equations. *Journal of Computational and Applied Mathematics*, 236(6):1531 – 1542.

Boyd, S. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press.

Buhmann, M. D. (2003). *Radial basis functions: theory and implementations*, volume 12. Cambridge university press.

Burfeindt, M. J., Colgan, T. J., Mays, R. O., Shea, J. D., Behdad, N., Van Veen, B. D., and Hagness, S. C. (2012). Mri-derived 3-d-printed breast phantom for microwave breast imaging validation. *IEEE antennas and wireless propagation letters*, 11:1610–1613.

Chatterjee, A. (2000). An introduction to the proper orthogonal decomposition. *Current science*, 78(7):808–817.

Chen, C.-T. (1995). *Linear system theory and design*. Oxford University Press, Inc.

Chiprout, E. and Nakhla, M. S. (1994). Asymptotic waveform evaluation. In *Asymptotic Waveform Evaluation*, pages 15–39. Springer.

Clark, I. and Harper, W. (2000). Practical geostatistics: Ecosse north american llc. *Columbus, OH*.

Coggon, J. (1971). Electromagnetic and electrical modeling by the finite element method. *Geophysics*, 36(1):132–155.

Colgan, T. J., Hagness, S. C., and Van Veen, B. D. (2015). A 3-d level set method for microwave breast imaging. *IEEE Transactions on Biomedical Engineering*, 62(10):2526–2534.

Conn, A. R., Scheinberg, K., and Vicente, L. N. (2009). *Introduction to derivative-free optimization*. SIAM.

Coulomb, J.-L. and Sabonnadière, J.-C. (1985). *CAO en électrotechnique*. Hermes.

Courant, R. et al. (1943). Variational methods for the solution of problems of equilibrium and vibrations. *Bull. Amer. Math. Soc*, 49(1):1–23.

Cullum, J., Ruehli, A., and Zhang, T. (2000). A method for reduced-order modeling and simulation of large interconnect circuits and its application to peec models with retardation. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 47(4):261–273.

Currie, I. G. (2012). *Fundamental mechanics of fluids*. CRC Press.

Delinchant, B., Lahaye, D., Wurtz, F., and Coulomb, J.-L. (2013). Manifold mapping optimization with or without true gradients. *Mathematics and Computers in Simulation*, 90:256–265.

Demerdash, N. and Nehl, T. (1979). An evaluation of the methods of finite elements and finite differences in the solution of nonlinear electromagnetic fields in electrical machines. *IEEE Transactions on Power Apparatus and Systems*, (1):74–87.

Ding, J. and Zhou, A. (2007). Eigenvalues of rank-one updated matrices with some applications. *Applied Mathematics Letters*, 20(12):1223–1226.

Elsherbeni, A. Z. and Demir, V. (2016). *The finite-difference time-domain method for electromagnetics with MATLAB simulations*. The Institution of Engineering and Technology.

Enns, D. F. (1984). Model reduction with balanced realizations: An error bound and a frequency weighted generalization. In *Decision and Control, 1984. The 23rd IEEE Conference on*, pages 127–132. IEEE.

Fairman, F. W. (1998). *Linear control theory: the state space approach*. John Wiley & Sons.

Fennell, T., Deen, P., Wildes, A., Schmalzl, K., Prabhakaran, D., Boothroyd, A., Aldus, R., McMorrow, D., and Bramwell, S. (2009). Magnetic coulomb phase in the spin ice ho2ti2o7. *Science*, 326(5951):415–417.

Feynman, R. P., Leighton, R. B., and Sands, M. (2013). *The Feynman Lectures on Physics, Desktop Edition Volume I*, volume 1. Basic books.

Feynman, R. P., Leighton, R. B., and Sands, M. L. (1964). *The Feynman Lectures on Physics: electromagnetism and matter*, volume 2. Addison Wesley Publishing Company.

Fletcher, R. (2013). *Practical methods of optimization*. John Wiley & Sons.

Franco, D. G. B. and Steiner, M. T. A. (2017). New strategies for initialization and training of radial basis function neural networks. *IEEE Latin America Transactions*, 15(6):1182–1188.

Freschi, F., Gruosso, G., and Repetto, M. (2006). Unstructured peec formulation by dual discretization. *IEEE microwave and wireless components letters*, 16(10):531–533.

Freund, R. W. (2000). Krylov-subspace methods for reduced-order modeling in circuit simulation. *Journal of Computational and Applied Mathematics*, 123(1):395–421.

Freund, R. W. (2011a). The sprim algorithm for structure-preserving order reduction of general rcl circuits. In *Model reduction for circuit simulation*, pages 25–52. Springer.

Freund, R. W. (2011b). The sprim algorithm for structure-preserving order reduction of general rcl circuits. In *Model reduction for circuit simulation*, pages 25–52. Springer.

Friedland, B. (2012). *Control system design: an introduction to state-space methods*. Courier Corporation.

Garcia, P. and Webb, J. (1990). Optimization of planar devices by the finite element method. *IEEE transactions on microwave theory and techniques*, 38(1):48–53.

Gershgorin, S. A. (1931). Uber die abgrenzung der eigenwerte einer matrix. *Bulletin de l'Academie des Sciences de l'URSS*, (6):749–754.

Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Artificial Intelligence. Addison-Wesley.

Golub, G. H. and Van Loan, C. F. (2012). *Matrix computations*, volume 3. JHU Press.

Gustavsen, B. (2006). Improving the pole relocating properties of vector fitting. *IEEE Transactions on Power Delivery*, 21(3):1587–1592.

Gustavsen, B. and Semlyen, A. (1999). Rational approximation of frequency domain responses by vector fitting. *IEEE Transactions on power delivery*, 14(3):1052–1061.

Gustavsen, B. and Semlyen, A. (2001). Enforcing passivity for admittance matrices approximated by rational functions. *IEEE Transactions on Power Systems*, 16(1):97–104.

Hayt, W. H. and Buck, J. A. (1981). *Engineering electromagnetics*, volume 6. McGraw-Hill New York.

Henneron, T. and Clenet, S. (2014). Model order reduction of non-linear magnetostatic problems based on pod and dei methods. *IEEE Transactions on Magnetics*, 50(2):33–36.

Hespanha, J. P. (2009). *Linear systems theory*. Princeton university press.

Isaaks, E. H. et al. (1989). *Applied geostatistics*. Number 551.72 I86. Oxford University Press.

Jamil, M. and Yang, X.-S. (2013). A literature survey of benchmark functions for global optimisation problems. *International Journal of Mathematical Modelling and Numerical Optimisation*, 4(2):150–194.

Jin, J.-M. (2014). *The finite element method in electromagnetics*. John Wiley & Sons.

Jin, J.-M. (2015). *The finite element method in electromagnetics*. John Wiley & Sons.

Jolliffe, I. (2002). *Principal component analysis*. Wiley Online Library.

Jones, D. R., Schonlau, M., and Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492.

Knowles, J. (2006). Parego: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10(1):50–66.

Kuwabara, Y., Wada, K., Guichon, J. M., Schanen, J. L., and Roudet, J. (2016). Implementation and performance of a current sensor for a laminated bus bar. In *2016 IEEE Energy Conversion Congress and Exposition (ECCE)*, pages 1–7.

Lanczos, C. (1950). *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*. United States Governm. Press Office Los Angeles, CA.

Laub, A. (1980). Efficient multivariable frequency response computations. In *1980 19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, number 19, pages 39–41.

Laub, A., Heath, M., Paige, C., and Ward, R. (1987). Computation of system balancing transformations and other applications of simultaneous diagonalization algorithms. *IEEE Transactions on Automatic Control*, 32(2):115–122.

Lee, J. M. (2003). Smooth manifolds. In *Introduction to Smooth Manifolds*, pages 1–29. Springer.

Li, M., Li, C., Ong, C.-J., and Tang, W. (2012). A novel multilevel matrix compression method for analysis of electromagnetic scattering from pec targets. *IEEE Transactions on Antennas and Propagation*, 60(3):1390–1399.

Lophaven, S. N., Nielsen, H. B., and Søndergaard, J. (2002). Dace-a matlab kriging toolbox, version 2.0. Technical report.

Luenberger, D. G. D. G. (1979). Introduction to dynamic systems; theory, models, and applications. Technical report.

Maxwell, J. C. (1881). *A treatise on electricity and magnetism*, volume 1. Clarendon press.

Montgomery, D. C. (2017). *Design and analysis of experiments*. John Wiley & Sons.

Moore, B. (1981). Principal component analysis in linear systems: Controllability, observability, and model reduction. *IEEE transactions on automatic control*, 26(1):17–32.

Nguyen, T. S. (2012). *Réduction de modèles issus de la méthode PEEC pour la modélisation électromagnétique des interconnexions électriques*. PhD thesis, Grenoble.

Nitsch, J., Gronwald, F., and Wollenberg, G. (2009). *Radiating nonuniform transmission-line systems and the partial element equivalent circuit method*. John Wiley & Sons.

Odabasioglu, A., Celik, M., and Pileggi, L. T. (1997). Prima: passive reduced-order interconnect macromodeling algorithm. In *Proceedings of the 1997 IEEE/ACM international conference on Computer-aided design*, pages 58–65. IEEE Computer Society.

Oehlert, G. W. (2010). *A first course in design and analysis of experiments*.

Oppenheim, A., Willsky, A., and Nawab, S. (1997). *Signals and Systems*. Prentice-Hall signal processing series. Prentice Hall.

Paixão, C. A. and Coelho, F. C. (2015). Matrix compression methods. Technical report, PeerJ PrePrints.

Panzer, H., Hubele, J., Eid, R., and Lohmann, B. (2009). Generating a parametric finite element model of a 3d cantilever timoshenko beam using matlab. *Tech. reports on aut. control, Inst. Aut. Control, TU München*.

Panzer, H., Mohring, J., Eid, R., and Lohmann, B. (2010). Parametric model order reduction by matrix interpolation. *at-Automatisierungstechnik Methoden und Anwendungen der Steuerungs-, Regelungs-und Informationstechnik*, 58(8):475–484.

Peherstorfer, B. and Willcox, K. (2016). Data-driven operator inference for nonintrusive projection-based model reduction. *Computer Methods in Applied Mechanics and Engineering*, 306:196–215.

Polycarpou, A. C. (2005). Introduction to the finite element method in electromagnetics. *Synthesis Lectures on Computational Electromagnetics*, 1(1):1–126.

Prando, G. and Chiuso, A. (2015). Model reduction for linear bayesian system identification. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 2121–2126. IEEE.

Press, W. H. (2007). *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press.

Qiang, Z., Ya-bo, Z., and Shu, M. (2011). Effective global searching method based on radial basis function. In *2011 International Conference on Electric Information and Control Engineering*.

Rao, S. S. (2009). *Engineering optimization: theory and practice*. John Wiley & Sons.

Recipes, N. (2007). The art of scientific computing. *University of Cambridge (http://www. nr. com)*.

Roudet, J., Clavel, E., Guichon, J.-M., and Schanen, J.-L. (2007). Inca3d a cad tool for stray elements computation in electrical engineering. In *ECPE Seminar:«Virtual Prototyping in Power Electronics*.

Rudolph, D. and Stitt, G. (2015). An interpolation-based approach to multi-parameter performance modeling for heterogeneous systems. In *2015 IEEE 26th International Conference on Application-specific Systems, Architectures and Processors (ASAP)*, pages 174–180. IEEE.

Ruehli, A., Antonini, G., and Jiang, L. (2017). *Circuit Oriented Electromagnetic Modeling Using the Peec Techniques*. John Wiley & Sons.

Ruehli, A. E. (1974). Equivalent circuit models for three-dimensional multiconductor systems. *IEEE Transactions on Microwave theory and techniques*, 22(3):216–221.

Ruhe, A. (1994). The rational krylov algorithm for nonsymmetric eigenvalue problems. iii: Complex shifts for real matrices. *BIT Numerical Mathematics*, 34(1):165–176.

Saad, Y. (2003). *Iterative methods for sparse linear systems*. SIAM.

Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989). Design and analysis of computer experiments. *Statistical science*, pages 409–423.

Sarma, M. (1976). Magnetostatic field computation by finite element formulation. *IEEE Transactions on Magnetics*, 12(6):1050–1052.

Schneider, J. B. (2010). Understanding the finite-difference time-domain method. *School of electrical engineering and computer science Washington State University.–URL: http://www. Eecs. Wsu. Edu/˜ schneidj/ufdtd/(request data: 29.11. 2012)*.

Seifi, H. and Sepasian, M. S. (2011). *Electric power system planning: issues, algorithms and solutions*. Springer Science & Business Media.

Semlyen, A. and Gustavsen, B. (2000). Vector fitting by pole relocation for the state equation approximation of nonrational transfer matrices. *Circuits, Systems and Signal Processing*, 19(6):549–566.

Siau, J. (2016). *Unstructured PEEC formulations considering resistive, inductive and capacitive effects for power electronics*. PhD thesis, Université Grenoble Alpes.

Silvester, P. (1969). Finite element solution of homogeneous waveguide problems. *Alta Frequenza*, 38(Speciale):313–317.

Simic, D., Bäuml, T., and Pirker, F. (2008). Modeling and simulation of different hybrid electric vehicles in modelica using dymola. In *Proceedings of International Conference on Advances in Hybrid Powertrains, Conference on Advances in Hybrid Powertrains*.

Spivak, M. (1965). *Calculus on manifolds: a modern approach to classical theorems of advanced calculus*. Westview Press.

Strang, G. (2007). *Computational science and engineering*, volume 1. Wellesley-Cambridge Press Wellesley.

Strang, G. (2011). Introduction to linear algebra.

Strang, G. and Aarikka, K. (1986). *Introduction to applied mathematics*, volume 16. Wellesley-Cambridge Press Wellesley, MA.

Stykel, T. (2002). Analysis and numerical solution of generalized lyapunov equations. *Institut für Mathematik, Technische Universität, Berlin*.

Taflove, A. and Hagness, S. C. (2005). *Computational electrodynamics: the finite-difference time-domain method*. Artech house.

Trefethen, L. N. and Bau III, D. (1997). *Numerical linear algebra*, volume 50. Siam.

Volakis, J. L., Chatterjee, A., and Kempel, L. C. (1998). *Finite element method electromagnetics: antennas, microwave circuits, and scattering applications*, volume 6. John Wiley & Sons.

Walecka, J. D. (2007). *Introduction to general relativity*. World Scientific Publishing Co Inc.

Wang, X., Zhou, W., and Dou, R. (2012). Analysis of harmonic current in permanent magnet synchronous motor and its effect on motor torque. *Journal of Electromagnetic Analysis and Applications*, 4(01):15.

Warwick, K. and Craddock, R. (1996). An introduction to radial basis functions for system identification. a comparison with other neural network methods. In *Proceedings of 35th IEEE Conference on Decision and Control*, volume 1, pages 464–469 vol.1.

Wendland, H. (2004). *Scattered data approximation*, volume 17. Cambridge university press.

Willcox, K. and Peraire, J. (2002). Balanced model reduction via the proper orthogonal decomposition. *AIAA journal*, 40(11):2323–2330.

Xia, B., Ren, Z., and Koh, C.-S. (2014). Utilizing kriging surrogate models for multi-objective robust optimization of electromagnetic devices. *IEEE Transactions on Magnetics*, 50(2):693–696.