Utrecht University

Faculty of Science

Truncation based model order reduction techniques and their application to digital audio filters

Bachelor Thesis Mathematics

David D. de Best (5647592)

Supervisor:
Dr. Tristan van Leeuwen

June 1, 2020

Abstract

Model order reduction aims to reduce the order of high-order dynamical systems while preserving most of their input-output behaviour. In this thesis, we will give a short introduction to model order reduction and examine two methods in detail that are considered truncation techniques, balanced truncation and modal truncation. We will discuss some of their properties and apply these methods to a real-life scenario by trying to reduce the order of several digital audio filters. These filters are models that process a digital audio signal to generate varying sounds. Next, we will compare the results of different reductions in terms of a computable error bound on the approximations. We will see that for our specific type of filter, balanced truncation seems to be able to provide the best reductions, where we can sometimes find an accurate approximation with an order reduced by more than 50%.

Contents

1 Introduction

In computational science, many phenomena can be modelled using dynamical systems that may contain a very large of amount of variables or equations. These systems are models of a high order and therefore require a lot of computation to study their behaviour. A common example is a circuit model of a particular electrical circuit that might consist of a very large number of elements. This circuit can be modeled using a large amount of equations, creating a system of a very high order. This could easily be a system of $\sim 10^5$ equations. Calculations become very costly, and so it would be useful to approximate this model with a model of a much lower order.

The aim of *model order reduction (MOR)* is to approximate a certain high-order model by replacing it with a model of a lower order, trying to assure that the reduced-order model produces an output that is similar to the output of the original system under the same input. This allows the behaviour of models to be studied more effectively, while preserving a reliable outcome. Model order reduction often tries to capture certain essential properties of a system in an automatic way. Two main categories of methods are *projection* techniques and *truncation* techniques, and there exist methods that combine methods from these categories or fall outside of these categories.

The two truncation techniques that will be discussed in this thesis are *balanced truncation* and *modal truncation*. Balanced truncation was published by Moore[1] in 1981 and was developed in the field of systems and control theory. In 1984, Glover[2] published the method of Hankel-norm reduction, belonging to the same category of techniques. These methods are based on the idea that the largest singular values of a system are important to the output of a system. Modal truncation differs from these, and produces an approximation by using the dominant modes of a system in the reduced model.

In the category of projection techniques fall methods that usually project the model to a certain space of a lower dimension. One of the first methods was *Asymptotic Waveform Evaluation*. There exist also several *Krylov-subspace* methods and another method is the *Proper Orthogonal Decomposition method*. These are not discussed in this thesis, but they are summarized in [3] and [4]. There exist many MOR approaches, of which some have a more general use and others are useful for rather specific cases.

We will go more in-depth on the truncation techniques and we will apply them to several digital audio filters. Digital audio filters used in audio production and design are often represented as a dynamical system. This application often requires fast computation of the effect of a filter on a certain input and therefore it is a subject that might benefit greatly from reducing the order of these filters.

The aim of this thesis is to give an introduction to model order reduction by discussing two basic truncation techniques in detail and applying them to real-world examples to show what potential reductions of the model order are possible. In this section we have given a very short introduction to the field of model order reduction. In section 2, we cover some basic concepts and representations of models used in MOR and we present the theory of the two MOR methods in detail. We also discuss these methods in terms of stability and error bounds and we list some methods that improve or alterate the based truncation techniques. In section 3, we examine the set-up of the experiments and some theory necessary to put the MOR techniques in practice. We present the results of the experiments in section 4 and compare them, and we conclude in section 5.

2 Model order reduction

This section will cover the necessary theory to put truncation MOR techniques in practice. In the first part, we will look at some representations of models which will be used to apply the techniques. In the second part, we will describe the process of balanced truncation and we will do the same for modal truncation in the third part. Finally, we will discuss these methods in terms of stability, error analysis and computational obstacles.

2.1 Model representations

In this section we will look at different representations for the models that will represent an audio filter in section 3. These representations allow for different types of reduction that can be applied. We will also see how MOR transforms these system in the general case. The first three parts are based on [4] and [5].

2.1.1 State-space models

Many models in computational science are described by a system of ordinary differential equations in the time domain. Let $\mathbf{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ and $\mathbf{g} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^p$ be two functions, then such a dynamical system is typically of the form

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)),$$
$$\mathbf{y}(t) = \mathbf{g}(\mathbf{x}(t), \mathbf{u}(t)).$$

In this system, $\mathbf{u} \in \mathbb{R}^m$ is the input, $\mathbf{y} \in \mathbb{R}^p$ the output and $\mathbf{x} \in \mathbb{R}^n$ is a *state* vector. We often assume the initial conditions where $\mathbf{u}(0) = 0, \quad \mathbf{x}(0) = 0$ and therefore $\mathbf{y}(0) = 0$. This formulation of a dynamical system is called a *state-space representation.* In a state-space representation, the output depends on the state variables of which the values change through time, dependent on the values they have at a certain point in time and the value of the input. It is a very convenient formulation when we have a model that is described by a high-order differential equation, as this form allows us to replace that model with a system of first-order differential equations [6]. When $\mathbf{f}$ and $\mathbf{g}$ are linear functions, this system is written in a more simple way using matrices as

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t), \tag{2.1a}$$
$$\mathbf{y}(t) = \mathbf{C}^T\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t). \tag{2.1b}$$

Here $\mathbf{A} \in \mathbb{R}^{n\times n}$ is often called the state matrix, which must be invertible for the equation to have a unique solution. Matrices $\mathbf{B} \in \mathbb{R}^{n\times m}, \mathbf{C} \in \mathbb{R}^{n\times p}$ and $\mathbf{D} \in \mathbb{R}^{p\times m}$ are respectively the input, output and feedthrough matrix. This feedthrough matrix is the zero matrix when there is no direct feedthrough of the input in the system. All matrices can be time-dependent, in which case the model is called a time-varying system, or time-independent, in which case we speak of a time-invariant system. The systems under considerations will all be *linear time-invariant systems (LTI)*, so the formulation of (2.1) will be used. If $m > 1$ and $p > 1$, system (2.1) is a *multi-input, multi-output (MIMO)* system. When $m = p = 1$, the system is termed *single-input, single-output (SISO)* and it is formulated as

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{b}u(t), \tag{2.2a}$$
$$y(t) = \mathbf{c}^T\mathbf{x}(t) + du(t). \tag{2.2b}$$

Here we use vectors $\mathbf{b}, \mathbf{c} \in \mathbb{R}^n$ and values $d, u, y \in \mathbb{R}$.

The *stability* of a system is determined by matrix $\mathbf{A}$. This property assures that the output signal of the system is bounded in the time domain. We call a system and the corresponding matrix $\mathbf{A}$ stable if its eigenvalues λ_i have non-positive real parts and if all λ_i are simple, for which $\mathrm{Re}(\lambda_i) = 0$, giving us the following definition

Definition 2.1.1. A linear time-invariant state-space system formulated as (2.1) is termed stable if and only if for all eigenvalues λ_i of $\mathbf{A}$
(a) $\mathrm{Re}(\lambda_i) \leq 0$
(b) if $\mathrm{Re}(\lambda_i) = 0$ then $\lambda_i \neq \lambda_j$ for all j with $j \neq i$

If a certain system is stable, we would like this property to be preserved during order reduction.

2.1.2 Similarity transformation

We often want to do a linear change of coordinates of the state vector $\mathbf{x}$. This is done using a *similarity transformation* on the state-space system. Let the matrix $\mathbf{T}$ represent the one-to-one linear transformation, then applying it to the state vector gives

$$\mathbf{x} = \mathbf{T}\tilde{\mathbf{x}} \quad \text{and} \quad \tilde{\mathbf{x}} = \mathbf{T}^{-1}\mathbf{x},$$

as the inverse of a one-to-one linear transformation always exists. By substituting $\mathbf{x} = \mathbf{T}\tilde{\mathbf{x}}$ in system (2.1), we obtain the system

$$\begin{aligned}\mathbf{T}\dot{\tilde{\mathbf{x}}}(t) &= \mathbf{A}\mathbf{T}\tilde{\mathbf{x}}(t) + \mathbf{B}\mathbf{u}(t),\\ \mathbf{y}(t) &= \mathbf{C}^T\mathbf{T}\tilde{\mathbf{x}}(t) + \mathbf{D}\mathbf{u}(t).\end{aligned}$$

Now left-multiply the first equation with $\mathbf{T^{-1}}$, resulting in

$$\begin{aligned}\dot{\tilde{\mathbf{x}}}(t) &= \mathbf{T}^{-1}\mathbf{A}\mathbf{T}\tilde{\mathbf{x}}(t) + \mathbf{T}^{-1}\mathbf{B}\mathbf{u}(t),\\ \mathbf{y}(t) &= \mathbf{C}^T\mathbf{T}\tilde{\mathbf{x}}(t) + \mathbf{D}\mathbf{u}(t).\end{aligned}$$

This means we can always perform a similarity transformation on a state-space system by making the following substitutions in (2.1)

$$\dot{\mathbf{x}}(t) = \widetilde{\mathbf{A}}\mathbf{x}(t) + \widetilde{\mathbf{B}}\mathbf{u}(t), \tag{2.3a}$$
$$\mathbf{y}(t) = \widetilde{\mathbf{C}}^T\mathbf{x}(t) + \widetilde{\mathbf{D}}\mathbf{u}(t), \tag{2.3b}$$

where

$$\begin{aligned}\widetilde{\mathbf{A}} &= \mathbf{T}^{-1}\mathbf{A}\mathbf{T},\\ \widetilde{\mathbf{B}} &= \mathbf{T}^{-1}\mathbf{B},\\ \widetilde{\mathbf{C}}^T &= \mathbf{C}^T\mathbf{T},\\ \widetilde{\mathbf{D}} &= \mathbf{D}.\end{aligned}$$

A very convenient feature of the similarity transformation is the preservation of stability, because the eigenvalues of $\mathbf{A}$ are eigenvalues of $\widetilde{\mathbf{A}}$ as well, which can be shown directly.

Theorem 2.1.2. *If the stable LTI state-space system (2.1) is transformed to system (2.3), using a linear transformation* $\mathbf{T}$*, the resulting system is also stable.*

Proof. Let $\mathbf{v}$ be an eigenvector of $\mathbf{A}$, λ its corresponding eigenvalue and $\widetilde{\mathbf{v}} = \mathbf{T}^{-1}\mathbf{v}$ the transformed eigenvector. Then we find

$$\widetilde{\mathbf{A}}\widetilde{\mathbf{v}} = (\mathbf{T}^{-1}\mathbf{A}\mathbf{T})(\mathbf{T}^{-1}\mathbf{v}) = \mathbf{T}^{-1}\mathbf{A}\mathbf{v} = \mathbf{T}^{-1}\lambda\mathbf{v} = \lambda\widetilde{\mathbf{v}}.$$

Therefore, an eigenvalue of $\widetilde{\mathbf{A}}$ is an eigenvalue of $\mathbf{A}$ as well, and system (2.3) is stable by definition (2.1.1). □

2.1.3 General order reduction

The dimension n of state variable $\mathbf{x}$ characterizes the complexity of the system and therefore n is considered the order of the model. The objective of model order reduction (MOR) is to replace the problem with one of a reduced order r that produces a similar output to the output of the original system under the same input. There exist various measures for a certain output being similar to the original output. These measures are often the basis for a particular reduction method. For a linear time-invariant system, order reduction comes down to reducing the matrix $\mathbf{A}$ to $\hat{\mathbf{A}} \in \mathbb{R}^{r\times r}$, which means the matrices $\mathbf{B}$ and $\mathbf{C}$ are reduced to $\hat{\mathbf{B}} \in \mathbb{R}^{r\times m}$ and $\hat{\mathbf{C}} \in \mathbb{R}^{r\times p}$. This will result in the following system

$$\dot{\tilde{\mathbf{x}}}(t) = \hat{\mathbf{A}}\tilde{\mathbf{x}}(t) + \hat{\mathbf{B}}\mathbf{u}(t), \tag{2.4a}$$

$$\tilde{\mathbf{y}}(t) = \hat{\mathbf{C}}^T\tilde{\mathbf{x}}(t) + \mathbf{D}\mathbf{u}(t), \tag{2.4b}$$

with state variable $\tilde{\mathbf{x}} \in \mathbb{R}^r$ and output $\tilde{\mathbf{y}} \in \mathbb{R}^p$.

2.1.4 Transfer function

It is often useful to analyse a dynamical system by transforming it to the frequency domain from the time domain. In this section and the next, we look at a representation of a model in the frequency domain using a *transfer function*, as described by Antoulas [3]. Using the well-known Laplace transform, a system of differential equations can be transformed into algebraic equations. This (unilateral) Laplace transform is defined as

$$F(s) = \int_0^\infty f(t)e^{-st}dt,$$

where a real function $f(t), \quad t \geq 0$, is transformed to the complex-valued function $F(s)$, where $s = \sigma + \omega i$ is the frequency parameter. Values of s where $\sigma = 0$ and $\omega \geq 0$ are referred to as the *angular frequency* or *radian frequency*, which will represent our input signals in the frequency domain.

The linearity of the Laplace transform is easily shown, so when applied to the state-space model (2.1) the result is the following representation in the frequency domain:

$$s\mathbf{X}(s) = \mathbf{A}\mathbf{X}(s) + \mathbf{B}\mathbf{U}(s), \tag{2.5a}$$

$$\mathbf{Y}(s) = \mathbf{C}^T\mathbf{X}(s) + \mathbf{D}\mathbf{U}(s). \tag{2.5b}$$

The state $\mathbf{x}(t)$, input $\mathbf{u}(t)$ and output $\mathbf{y}(t)$ are transformed to vectors $\mathbf{X}(s), \mathbf{U}(s)$ and $\mathbf{Y}(s)$ respectively, with dimensions equal to the original vector. Here, we assume the zero-conditions $\mathbf{x}(0) = 0$, $\mathbf{u}(0) = 0$ and $\mathbf{y}(0) = 0$. Note that this is a system of purely algebraic equations, where the state variable $\mathbf{X}(s)$ can be eliminated to acquire the relation

$$\mathbf{Y}(s) = (\mathbf{C}^T(s\mathbf{I}_n - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D})\mathbf{U}(s),$$

which is possible since $\mathbf{A}$ was an invertible matrix. We now define a matrix-valued transfer function $\mathbf{H}(s) \in \mathbb{C}^{p\times m}$ as

$$\mathbf{H}(s) = \mathbf{C}^T(s\mathbf{I}_n - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D}. \tag{2.6}$$

This transfer function describes a direct relation between the input and output of the system in the frequency domain, which means the complete behaviour of the system can be analysed from this transfer function.

2.1.5 Pole-residue representation

We would like to expand the transfer function (2.6) as

$$\mathbf{H}(s) = \sum_{i=1}^{n} \frac{R_i}{s - p_i} + \mathbf{D}.$$

Here $p_i \in \mathbb{C}$ is called a pole and $R_i \in \mathbb{C}$ its corresponding residue. To achieve this representation, we will look only at SISO systems, so we examine the scalar $H(s) \in \mathbb{C}$. An extension to MIMO systems is given in [7]. To produce the pole-residue form, the eigenvalues $\lambda_i \in \mathbb{C}, i = 1, ..., n$, of $\mathbf{A}$ are needed. Each eigenvalue has a corresponding right and left eigenvector, $\mathbf{v}_i \in \mathbb{C}^n$ and $\mathbf{w}_i \in \mathbb{C}^n$ respectively.
We assume that $\mathbf{A}$ is a non-defective matrix and therefore the right and left eigenvectors can be scaled using factor ϕ so that $(\bar{\phi}\mathbf{w}_i^*)(\phi\mathbf{v}_i) = 1$, given that $\phi^2(\mathbf{w}_i^*\mathbf{v}_i) = 1$. Here, $\bar{\phi}$ denotes the complex conjugate of ϕ and $\mathbf{w}_i^*$ the conjugate transpose of $\mathbf{w}_i$. It can also be shown that left and right eigenvectors of different eigenvalues are orthogonal, so $\mathbf{w}_i^*\mathbf{v}_i = 0, i \neq j$. Let $\mathbf{V}$ and $\mathbf{W}$ have the right and left eigenvectors as columns, and $\mathbf{W}^*\mathbf{A}\mathbf{V} = \mathbf{\Lambda} = \text{diag}(\lambda_1, ..., \lambda_n)$. It follows that $\mathbf{W}^*\mathbf{V} = \mathbf{I}_n$. By doing a similarity transformation by setting $\mathbf{x} = \mathbf{V}\tilde{\mathbf{x}}$, we find the transformed system

$$\dot{\tilde{\mathbf{x}}}(t) = \mathbf{\Lambda}\tilde{\mathbf{x}}(t) + \mathbf{W}^*\mathbf{b}u(t), \tag{2.7a}$$

$$\tilde{y}(t) = \mathbf{c}^T\mathbf{V}\tilde{\mathbf{x}}(t) + du(t). \tag{2.7b}$$

The transfer function of this system is clearly

$$\mathbf{H}(s) = \mathbf{c}^T\mathbf{V}(s\mathbf{I}_n - \mathbf{\Lambda})^{-1}\mathbf{W}^*\mathbf{b} + d. \tag{2.8}$$

By doing the matrix multiplications, it can easily be shown that this transfer function is

$$\mathbf{H}(s) = \sum_{i=1}^{n} \frac{R_i}{s - \lambda_i} + d, \tag{2.9}$$

where $R_i = (\mathbf{c}^T\mathbf{v}_i)(\mathbf{w}_i^*\mathbf{b})$. This is exactly the formulation we were looking for and it will be the basis for the modal truncation method.
Since the eigenvalues λ_i of $\mathbf{A}$ are the poles of the transfer function, we can also formulate the stability of a state-space model in terms of the poles of its transfer function.

Definition 2.1.3. A linear time-invariant system with transfer function (2.9) is termed stable if and only if for all poles λ_i
(a) $Re(\lambda_i) \leq 0$
(b) if $Re(\lambda_i) = 0$ then $\lambda_i \neq \lambda_j$ for all j with $j \neq i$

2.2 Balanced Truncation

Balanced truncation or *Truncated Balanced Realization (TBR)* was first introduced by Moore[1]. This section will describe the process of balanced truncation as described in [3]. The method consists of a balancing part and then a truncation part. We are going to assume the state-space matrix $\mathbf{A}$ is stable. The first step is to find a *balancing transformation*. For this goal we need two concepts from control theory, the controllability and observability Gramian. We define a Gramian matrix in the following way [8].

Definition 2.2.1. A Gramian or Gram matrix G is defined as a Hermitian matrix of inner products of a set of vectors $v_1, ..., v_n$ in an inner product space V where $G = [\langle v_i, v_j \rangle]_{i,j=1}^n$.

We need the following theorem.

Theorem 2.2.2. *([8], 7.2.10) Let $v_1, ..., v_m$ be vectors in an inner product space V with innerproduct $\langle \cdot, \cdot \rangle$, and let $G = [\langle v_j, v_i \rangle]_{i,j=1}^m \in M_m$. Then*
(a) G is Hermitian and positive semidefinite
(b) G is positive definite if and only if the vectors $v_1, ..., v_m$ are linearly independent
(c) $\text{rank}(G) = \text{dimspan}\{v_1, ..., v_m\}$

Because of definition (2.2.1) and theorem (2.2.2), we can conclude that the controllability and observability Gramians are positive definite.
Given a linear system like (2.1), the finite and infinite controllability Gramian are defined as

$$\mathbf{W_c}(K) = \int_0^K e^{\mathbf{A}t}\mathbf{B}\mathbf{B}^T e^{\mathbf{A}^T t}dt, \quad \mathbf{W_c} = \int_0^\infty e^{\mathbf{A}t}\mathbf{B}\mathbf{B}^T e^{\mathbf{A}^T t}dt, \tag{2.10}$$

and the finite and infinite observability Gramian as,

$$\mathbf{W_o}(K) = \int_0^K e^{\mathbf{A}^T t}\mathbf{C}^T\mathbf{C}e^{\mathbf{A}t}dt, \quad \mathbf{W_o} = \int_0^\infty e^{\mathbf{A}^T t}\mathbf{C}^T\mathbf{C}e^{\mathbf{A}t}dt. \tag{2.11}$$

We use the exponential of a matrix here, described by the following power series [9]

$$e^{\mathbf{A}} = \sum_{m=0}^{\infty} \frac{\mathbf{A}^m}{m!}.$$

The infinite Gramians are the solutions to the following Lyapunov matrix equations

$$\mathbf{A}\mathbf{W}_c + \mathbf{W}_c\mathbf{A}^T = -\mathbf{B}\mathbf{B}^T, \tag{2.12a}$$
$$\mathbf{A}^T\mathbf{W}_o + \mathbf{W}_o\mathbf{A} = -\mathbf{C}^T\mathbf{C}. \tag{2.12b}$$

Using that $\mathbf{A}$ is stable and therefore has eigenvalues $\lambda_i, i = 1, 2, ..., n$ with negative real parts, we can diagonalize matrix $\mathbf{A}$ and determine the exponential by

$$e^{\mathbf{A}} = \mathbf{W}^* e^{\mathbf{\Lambda}}\mathbf{V},$$

where $\mathbf{\Lambda} = \text{diag}(\lambda_1, ..., \lambda_n)$ and where $\mathbf{W}^*$ denotes the conjugate transpose of $\mathbf{W}$. Now, $e^{\mathbf{\Lambda}} = \text{diag}(e^{\lambda_1}, ..., e^{\lambda_n})$, so if $t \to \infty$, we find that $e^{\mathbf{\Lambda}t} = 0_{n,n}$ and therefore $e^{\mathbf{A}t} = 0_{n,n}$ [9]. We can now show quite easily that the controllability Gramian is the solution to Lyapunov equation (2.12a) by doing

$$\mathbf{A}\mathbf{W}_c + \mathbf{W}_c\mathbf{A}^T = \int_0^\infty \mathbf{A}e^{\mathbf{A}t}\mathbf{B}\mathbf{B}^{\mathbf{T}}e^{\mathbf{A}^{\mathbf{T}}t}dt + \int_0^\infty e^{\mathbf{A}t}\mathbf{B}\mathbf{B}^{\mathbf{T}}e^{\mathbf{A}^{\mathbf{T}}t}\mathbf{A}^T dt =$$

$$\int_0^\infty \frac{d}{dt}(e^{\mathbf{A}t}\mathbf{B}\mathbf{B}^{\mathbf{T}}e^{\mathbf{A}^{\mathbf{T}}t})dt = 0 - \mathbf{B}\mathbf{B}^{\mathbf{T}} = -\mathbf{B}\mathbf{B}^{\mathbf{T}}.$$

This is done in a similar way for the observability Gramian. The goal is to find a basis transformation that introduces new coordinates $\tilde{\mathbf{x}}$ where $\mathbf{x} = {\mathbf{T_0}}^{-1}\tilde{\mathbf{x}}$. Here, the matrix $\mathbf{T_0}$ is the representation of the basis transformation. The two Gramians are transformed to

$$\widetilde{\mathbf{W_c}} = \mathbf{T_0}\mathbf{W_o}\mathbf{T_0^T} \text{ and } \widetilde{\mathbf{W_o}} = \mathbf{T_0^{-1}}\mathbf{W_o}\mathbf{T_0^{-T}}.$$

We need to make the transformed Gramians satisfy

$$\widetilde{\mathbf{W_c}} = \widetilde{\mathbf{W_o}} = \mathbf{\Sigma} = \text{diag}(\sigma_1, ..., \sigma_n).$$

Here, $\sigma_1, ..., \sigma_n$ are called the *Hankel Singular Values (HSV)*, defined as

$$\sigma_i = \sqrt{\lambda_k \langle \mathbf{W_c}, \mathbf{W_o} \rangle}, \quad i \in [1, ..., n],$$

the positive square roots of the eigenvalues of the inner product of the Gramians. If finding the balancing transformation $\mathbf{T_0}$ succeeds and $\mathbf{T_0}$ is applied to the system then the new system is called balanced.

The Gramians were positive definite matrices and therefore we can perform Cholesky factorisations to obtain upper triangular $\mathbf{U}$ and lower triangular $\mathbf{L}$ so that $\mathbf{W_c} = \mathbf{U}\mathbf{U}^T$ and $\mathbf{W_o} = \mathbf{L}\mathbf{L}^T$[8]. These matrices are real if $\mathbf{A}, \mathbf{B}, \mathbf{C}$ are real. By doing a singular value decomposition on the matrix $\mathbf{U}^T\mathbf{L}$, we obtain the Hankel singular values, following [10]. The singular value decomposition gives us

$$\mathbf{U}^T\mathbf{L} = \mathbf{Z}\mathbf{\Sigma}Y^*.$$

The matrix $\mathbf{\Sigma}$ contains all Hankel singular values on its diagonal and we can now construct the balancing transformation $\mathbf{T_0}$ by calculating

$$\mathbf{T_0} = \mathbf{U}\mathbf{Z}\mathbf{\Sigma}^{-\frac{1}{2}},$$

and

$$\mathbf{T_0}^{-1} = \mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{Y}^*\mathbf{L}^T.$$

So we have found a balancing transformation $\mathbf{T_0}$. This transformation is applied to the linear system (2.1), resulting in the balanced system

$$\begin{aligned} \dot{\mathbf{x}}(\mathbf{t}) &= \widetilde{\mathbf{A}}\mathbf{x}(\mathbf{t}) + \widetilde{\mathbf{B}}\mathbf{u}(\mathbf{t}) \\ \mathbf{y}(\mathbf{t}) &= \widetilde{\mathbf{C}}^T\mathbf{x}(\mathbf{t}) + \mathbf{D}\mathbf{u}(\mathbf{t}) \end{aligned}. \tag{2.13}$$

We now partition the matrix of Hankel singular values $\mathbf{\Sigma}$ so that

$$\mathbf{\Sigma} = \begin{pmatrix} \mathbf{\Sigma}_1 & 0 \\ 0 & \mathbf{\Sigma}_2 \end{pmatrix}.$$

As the dimensions of $\mathbf{\Sigma}_1$ will be the order of the reduced system, we can choose the dimensions of $\mathbf{\Sigma}_1$ and $\mathbf{\Sigma}_2$ to fit the desired order of the reduction. If the dimension of $\mathbf{\Sigma}_1$ is $r \times r$, the following matrices can be partitioned similarly:

$$\widetilde{\mathbf{A}} = \begin{pmatrix} \mathbf{A_{11}} & \mathbf{A_{12}} \\ \mathbf{A_{21}} & \mathbf{A_{22}} \end{pmatrix},$$

$$\widetilde{\mathbf{B}} = \begin{pmatrix} \mathbf{B_1} \\ \mathbf{B_2} \end{pmatrix},$$

$$\widetilde{\mathbf{C}} = \begin{pmatrix} \mathbf{C_1} \\ \mathbf{C_2} \end{pmatrix}.$$

A reduced system can now be constructed using matrices $\mathbf{A_{11}}, \mathbf{B_1}$ and $\mathbf{C_1}$:

$$\begin{aligned} \dot{\mathbf{x}}(\mathbf{t}) &= \mathbf{A_{11}}\mathbf{x}(\mathbf{t}) + \mathbf{B_1}\mathbf{u}(\mathbf{t}), \\ \mathbf{y}(\mathbf{t}) &= \mathbf{C}_1^T\mathbf{x}(\mathbf{t}) + \mathbf{D}\mathbf{u}(\mathbf{t}). \end{aligned} \tag{2.14}$$

2.3 Modal Truncation

The general idea of modal truncation is approximate a dynamical system by its dominant modes [7]. We will look at SISO systems, in [7] this method is extended to MIMO systems. We can obtain a reduced order model called the *modal equivalent* by projecting the state-space on a subspace spanned by the dominant modes. These dominant modes are certain eigenvectors and eigenvalues of state matrix $\mathbf{A}$. In practice, the objective of modal truncation is to replace the model with order n by a reduced model of order $r < n$ by replacing

the transfer function H in pole-residue notation, as defined in section 2.1.5, with a transfer function H_1 of r terms. This transfer function is therefore characterized as

$$H_1 = \sum_{i=1}^{r} \frac{R_i}{s - \lambda_i} + d, \tag{2.15}$$

with λ_i and R_i defined analogously to the original transfer function.
We need to make a selection of terms of the original transfer function that are used in the reduced function, so we need to choose a set of pairs (λ_i, R_i) of poles and residues. This can be done in different ways, one of which is based on the dominance of the pole, measured by $\frac{|R_i|}{|\mathrm{Re}(\lambda_i)|}$, as given by Rommes [7]. We choose the set of pairs with the largest dominance and produce a transfer function like (2.15). To find the corresponding state-space representation of the reduced system, a similarity transform with matrices $\mathbf{V}, \mathbf{W}$ has to be performed in the following way:

$$\widetilde{\mathbf{A}} = \mathbf{W}^* \mathbf{A} \mathbf{V},$$
$$\widetilde{\mathbf{b}} = \mathbf{W}^* \mathbf{b},$$
$$\widetilde{\mathbf{c}} = \mathbf{c} \mathbf{V}.$$

Matrix $\mathbf{V} \in \mathbb{C}^{n \times r}$ has the corresponding r right eigenvectors of the used eigenvalues and poles λ_i as its columns and matrix $\mathbf{W} \in \mathbb{C}^{n \times r}$ has the r left eigenvectors as its columns. We have now found a reduced system of the form (2.4). A notable advantage is that the poles of the reduced system were also poles of the original system.

2.4 Properties of the truncation methods

2.4.1 Stability

Both truncation methods have the useful property of preserving stability of the system. Balanced truncation requires the stability of the original system so that the Gramians are well defined. During the balancing transformation, the stability of the system is preserved because of theorem (2.1.2). Truncating the matrix $\widetilde{\mathbf{A}}$ to $\widetilde{\mathbf{A}}_{\mathbf{11}}$ guarantees the stability of the latter matrix, which is shown by Moore[1].
Applying modal truncation naturally preserves stability, as poles of the reduced transfer function were also poles of the original transfer function and therefore they adhere to the stability requirement in definition (2.1.3).

2.4.2 Error bound

A great advantage to both truncation methods is that they allow the computation of a particular error bound. When applying either form of MOR, one can use this information to select an appropriate reduction in terms of the possible size of the error in the approximation. It also allows us in this thesis to compare the results of these methods in the context of this error bound. The error bound is described using the $\mathbb{H}_\infty$-norm, defined as $\|\mathbf{H}\|_{\mathbb{H}_\infty} := \sup_{\boldsymbol{\omega} \in \mathbb{R}} \|\mathbf{H}(i\boldsymbol{\omega})\|_2$, where the $\|.\|_2$-norm is the matrix spectral norm. This means that the result of calculating the $\mathbb{H}_\infty$-norm is the largest singular value of the transfer function evaluated on the imaginary axis[10]. In a SISO system, this norm simply becomes $\|H\|_{\mathbb{H}_\infty} := \sup_{\boldsymbol{\omega} \in \mathbb{R}} |H(i\boldsymbol{\omega})|$.

If we define $\mathbf{H}$ as the transfer function of the original system and $\mathbf{H_1}$ as the transfer function of the reduced system, the error bound for balanced truncation given by Glover[2] holds

$$\|\mathbf{H} - \mathbf{H_1}\|_{\mathbb{H}_\infty} \leq 2 \cdot \sum_{i=r+1}^{n} \boldsymbol{\sigma}_i. \tag{2.16}$$

For modal truncation[11], the error in the reduced transfer function can be quantified as

$$\|\mathbf{H} - \mathbf{H_1}\|_{\mathbb{H}_\infty} \leq \sum_{i=r+1}^{n} \frac{|R_i|}{|\mathrm{Re}(\lambda_i)|}. \tag{2.17}$$

2.4.3 Computation

When using systems of much higher order than the systems examined in this thesis, the truncation methods described above might become too computationally expensive. There exist some solutions to these problems listed here. Modal truncation as described in 2.3 requires a full eigenvalue decomposition of matrix $\mathbf{A}$ to produce a transfer function in pole-residue form. To calculate the error bound, we need the same eigenvalue decomposition. This might not be an option for very large scale systems, with order $n > 2000$ [3]. Rommes and Martins[12] present an approach for these systems by using the *subspace accelerated dominant pole algorithm (SADPA)*. They extend the method to MIMO systems in [13] via the *subspace accelerated MIMO dominant pole algorithm (SAMDP)*.

In balanced truncation, heavy computation is performed in solving the Lyapunov equations. This is usually done by making a Schur decomposition of matrix $\mathbf{A}$[4] and then some form of substitution in the obtained equation. An alternative is presented by Benner[14] using the matrix sign function. Another option is using *alternating direction implicit iteration (ADI)*, for example as done by Ellner and Wachspress[15].
A different approach is to calculate the Gramians directly from their definition instead of the Lyapunov equations. A method to do this is presented by Silveira and Philiphs[16] via *Poor Man's Truncated Balanced Realization (PMTBR)*.

3 Application to digital audio filters

In this section, we give a short introduction to digital audio filters in a very simple setting and we look at different concepts from section 2 in this context. Next, using the theory discussed in the previous section, we apply a reduction to the model order of a filter to get a better understanding of the methods. As digital filters are discrete-time systems in practice, some alterations to the truncation methods are needed and these will be discussed here. The information in the first two parts is based on [5].

3.1 Introduction to digital audio filters

In digital signal processing, filters are used to modify a digital signal, by transforming the current sample of the signal, possibly using surrounding samples of the input or output signal. In audio applications, these filters change the sound of audio signals mostly by adjusting the frequency, amplitude or phase or a mix of these. These calculations are often done in real-time, causing a need for good performance on the application of the filter. Therefore, filters of lower orders might be more appropriate in many cases. Here, we define a digital filter and discuss a few different types.
To start, we can define a digital signal in the following way.

Definition 3.1.1. *Digital signal* A discrete-time signal is a sequence of time-ordered real or complex numbers, that can be denoted as the real- or complex-valued function of an integer sample number $x(n), n \in \mathbb{Z}$.

It is useful to define the signal space $\mathcal{S} \subseteq \ell_2$ of all complex signals $x : \mathbb{N} \to \mathbb{C}$. A digital filter $\mathcal{T}$ will map an input signal x to an output signal y, which means the filter $\mathcal{T}$ can be regarded as an operator on the signal space $\mathcal{S}$.

Definition 3.1.2. *Digital audio filter* A digital audio filter $\mathcal{T}$ maps a signal x to a signal y, with $x, y \in \mathcal{S}$. The relation can be notated as

$$y(n) = \mathcal{T}_n(x(n)),$$

where $\mathcal{T}_n$ is the transformation of the input signal to a sample value at time n.

When the signals are defined as a sequence of real or complex numbers, the filter that is applied is considered a single-input, single-output filter. Alternatively, the input and output could be vector-valued and then the filter is multi-input, multi-output. We will mostly look at linear time-invariant SISO filters. These LTI filters are the only filters that preserve a signals frequency and they are therefore very important when processing audio[5].

Example 3.1.3. A simple example is a SISO filter defined by the difference equation

$$y(n) = x(n) + x(n-1),$$

where $x(n)$ is the signal input amplitude at sample n and $y(n)$ the signal ouput amplitude. This is a very simple low-pass filter, meaning that signals at low frequency are left unchanged by the filter, but the amplitude of signals with a high frequency is reduced. It is clearly linear and also time-invariant, because the map is the same independent of the current sample n. △

3.2 Adaptations for discrete time systems

We can set up a transfer function for example (3.1.3) like in section 2.1.4 by using the *unilateral z-transform*, that can be considered as the discrete-time equivalent of the Laplace-transform. This transfer function $H(z)$ is given by

$$H(z) = \frac{Y(z)}{X(z)},$$

where $Y(z)$ and $X(z)$ are the z-transform of the output and input signal, respectively. The z-transform is a linear operator that can be applied by using the following map

$$x(n - \Delta) \to X(n)z^{-\Delta}, \quad \Delta > 0.$$

This means we can find the transfer function of example 3.1.3 by applying the z-transform and solving for $Y(z)/X(z)$, so

$$\begin{aligned} Z\{y(n)\} &= Z\{x(n) + x(n-1)\}, \\ Z\{y(n)\} &= Z\{x(n)\} + Z\{x(n-1)\}, \\ Y(z) &= X(z) + X(z)z^{-1}, \\ H(z) &= \frac{Y(z)}{X(z)} = 1 + z^{-1}. \end{aligned}$$

We have now found a simple transfer function for our filter with a zero q at $q = 1$ and no poles.

Example (3.1.3) uses a simple *finite impulse response (FIR)* filter, as opposed to an *infinite impulse response (IIR)* filter. A FIR digital filter has an impulse response that is always zero after a finite number of samples. This impulse response is the output signal given by using an input signal with amplitude 1 at sample 0 and amplitude 0 at any other sample. These filters do not have any feedback loops and are also called non-recursive. On the other hand, an IIR filter has an impulse response that does not converge to zero due to the use of feedback and they are called recursive.
These IIR filters are often represented in the state-space form as defined in section 2. A discrete IIR state-space filter can be written as

$$\mathbf{x}(n+1) = \mathbf{A}\mathbf{x}(n) + \mathbf{b}u(n), \tag{3.1a}$$

$$y(n) = \mathbf{c}^T\mathbf{x}(n) + du(n). \tag{3.1b}$$

Here u is the input signal, $\mathbf{x}$ the vector of state variables and y the output signal.

A method to produce a state-space realization from a transfer function is given by Smith [5] by creating a state-space model in *controller canonical form*. We give a short summary here. Given a general IIR filter with transfer function

$$H(z) = \frac{b_0 + b_1 z^{-1} + ... + b_{n_b} z^{-n_b}}{1 + a_1 z^{-1} + ... + a_{n_a} z^{-n_a}},$$

we find the direct-path coëfficient d by doing one step of long division, which results in

$$H(z) = b_0 + \frac{(b_1 - b_0 a_1)z^{-1} + ... + (b_n - b_0 a_n)z^{-n}}{1 + a_1 z^{-1} + ... + a_{n_a} z^{-n_a}},$$

$$H(z) = b_0 + \frac{\beta_1 z^{-1} + ... + \beta_n z^{-n}}{1 + a_1 z^{-1} + ... + a_{n_a} z^{-n_a}}.$$

Here $n = \max(n_a, n_b)$ and $a_i = 0$ for $i > n_a$ $b_i = 0$ for $i > n_b$. The model in controller canonical form is now given by the matrices

$$\mathbf{A} = \begin{pmatrix} -a_1 & -a_2 & \cdots & -a_{n-1} & -a_n \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \quad \mathbf{c} = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \vdots \\ \beta_n \end{pmatrix}, \quad d = b_0.$$

We now have the possibility to transform a filter from a difference equation to a transfer function and from a transfer function to a state-space model. Using the pole-residue notation from section 2.1.5, we can transform a state-space model back to a transfer function. This will give us the necessary freedom when applying model order reduction to the filters.

For discrete systems, we need to make one important alteration to one of the reduction techniques. When applying balanced truncation to a discrete system, we need to find the discrete controllability and observability Gramian. These arise from the corresponding discrete Lyapunov equations [17]. The discrete controllability Gramian W_c and discrete observability Gramian W_o are the solutions of the equations

$$\mathbf{W}_c - \mathbf{A}^T\mathbf{W}_c\mathbf{A} = \mathbf{B}\mathbf{B}^T, \tag{3.2a}$$

$$\mathbf{A}^T\mathbf{W}_o\mathbf{A} - \mathbf{W}_o = -\mathbf{C}^T\mathbf{C}. \tag{3.2b}$$

Here, we must also note that the error bound for balanced truncation stated in section 2.4.2 also applies to discrete-time systems according to the work of Gu[18].

3.3 Implementation

All pythoncode used for implementation of the reduction techniques and to generate the data in this thesis can be found at: *https://github.com/DavidDdeBest/TruncationMORtoDAF*. To demonstrate the implementation of the truncation techniques applied to digital audio filters, we start with a small example.

Example 3.3.1. Consider an IIR filter with a transfer function $H(z)$ with poles $p \in \{-\frac{1}{2}, -2, -3, -4\}$ and zeros $q \in \{-1, \frac{1}{2}, 1, 2\}$. This means that

$$H(z) = \frac{(1+x)(-\frac{1}{2}+x)(-1+x)(-2+x)}{(1+2x)(1+\frac{1}{2}x)(1+\frac{1}{3}x)(1+\frac{1}{4}x)} = \frac{-\frac{1}{2}+x+x^2-\frac{5}{2}x^3+x^4}{1+\frac{47}{24}x+\frac{67}{42}x^2+\frac{41}{84}x^3+\frac{1}{12}x^4}.$$

We have a filter of order 4 and since every pole p is negative, this is a stable filter. This transfer function can be used to provide a state-space model in controller canonical form using the method described in section 3.2. The resulting system is described by the matrices

$$\mathbf{A} = \begin{pmatrix} -\frac{47}{24} & -\frac{67}{42} & -\frac{41}{84} & -\frac{1}{12} \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad \mathbf{C} = \begin{pmatrix} \frac{95}{96} \\ \frac{151}{84} \\ -\frac{379}{168} \\ \frac{25}{24} \end{pmatrix}, \quad \mathbf{D} = -\frac{1}{2}.$$

△

We now apply balanced truncation and modal truncation to example (3.3.1). Balanced truncation will provide us with a set of four Hankel singular values, that are used to calculate an error bound. These values are plotted below.

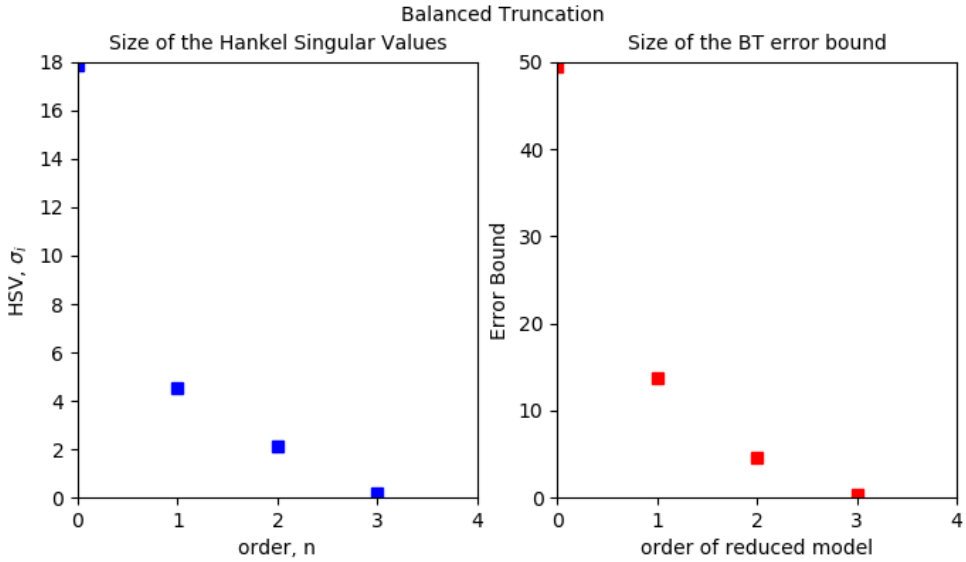

Figure 1: Balanced truncation applied to example (3.3.1). When we do not perform a reduction, so the model stays at order four, the error bound is naturally zero. There is no HSV plotted corresponding to this order as it does not exist.

As expected, the error bound corresponding to a reduction increases significantly with a reduction to a smaller order. We also use modal truncation to reduce the order of this filter. This provides us with a set of four poles that each have a dominance. These are used to calculate an error bound per possible order reduction.

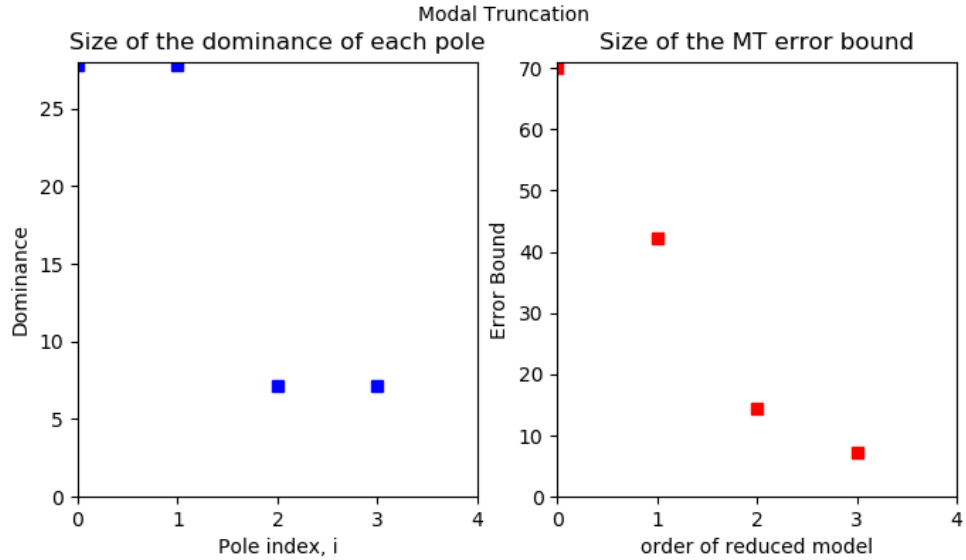

Figure 2: Modal truncation applied to example (3.3.1). When no reduction is done and the model order stays at order four, the error bound is naturally zero. There is no dominance of a pole plotted corresponding to this order as there does not exist a fifth pole.

We see that in this example that balanced truncation achieves significantly better results in terms of keeping a small error bound in comparison to modal truncation. Only when balanced truncation is used to reduce the order of this model to three, we seem to be able the find a reduced-order model with an error that is close to zero. Next, we will set up some more advanced examples.

3.4 Audio filter construction

In this section we describe the method of setting up filters of a certain order, and the motivation for this choice. We are going to construct an all-pass IIR dispersion filter. All-pass filters leave the amplitude unchanged for signals of any frequency. A dispersion filter changes the group delay of frequencies. The group delay may be interpreted as the time delay of the amplitude envelope of a sinusoid at frequency ω. These types of filters have their use in multiple applications, for example the synthesis of realistic percussive and piano string sounds[5].

Our goal is to compare the accuracy of reductions done by both methods to filters of different orders, especially at higher orders, where sizeable reductions can have a lot of benefits in terms of performance when using the filter. This means we want to design filters in a way that we can have influence on its order and so that we can make this order rather large.
Here, we use a method presented by Abel and Smith [19]. This method allows for an easy design of very high order filters in cascaded biquad form using any desired group delay function. An important feature is that we can influence the order of the resulting transfer function. This means that by using this method we can examine the truncation methods for different orders of filters. An alteration we make to the method is that we only use biquad sections that are stable, which is not garantueed in the original method. This restricts what types of group delay functions we can construct, but that is not a problem for our experiment, as the main goal is to just set up multiple filters of different orders.
We summarize the method here.

Given a certain group delay $\delta(\omega)$ that is dependent on the normalized radian frequency ω, we add a constant delay to $\delta(\omega)$ so that it integrates to a multiple of 2π. If $\int_0^\pi \delta(\omega)d\omega = 2k\pi$, the resulting transfer function will be k cascaded biquad factors and the order of the eventual system will be $2k$. A biquad is generally a two-pole, two-zero second-order filter, with a transfer function $G(z)$ of the form

$$G(z) = g\frac{1 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}.$$

The next step is to divide the group delay in 2π-area frequency bands by setting bounds $\omega_j, j \in \{0, 1, 2, ..., k\}$. This means determining a set of $\omega_j \in [0, \pi]$ so that $\int_{\omega_j}^{\omega_{j+1}} \delta(\omega)d\omega = 2\pi$, and $\sum_{j=0}^{k-1} \int_{\omega_j}^{\omega_{j+1}} \delta(\omega)d\omega = 2k\pi$. Here ω_0 and ω_k are set to 0 and π respectively.
We now fit a first-order all-pass section $H_i(z)$ to each band, by defining

$$H_j(z) = \frac{-\rho_j e^{-i\theta_j} + z^{-1}}{1 - \rho_j e^{i\theta_j} z^{-1}}.$$

This is a first-order filter with a pole p_j at $\rho_j e^{i\theta_j}$ and zero ζ_j at $\frac{1}{\rho_j} e^{i\theta_j}$. We also define

$$\theta_j = \frac{1}{2}(\omega_j + \omega_{j-1}), \qquad j \in \{1, 2, ..., k\}$$

and

$$\rho_j(\beta) = \eta_j - \sqrt{\eta_j^2 - 1}$$

where

$$\eta_j(\beta) = \frac{1 - \beta \cos\Delta_j}{1 - \beta}, \qquad \Delta_j = \frac{1}{2}(\omega_j - \omega_{j-1}).$$

β is a user-determined parameter, determining the 'smoothness' of the group delay approximation. We will use a value of 0.8, giving a fairly smooth curve.
Finally, these first-order sections are cascaded together with their complex conjugate $H'(z)_j$ to obtain a biquad filter with real coëfficients. The resulting transfer function is given by

$$H(z) = \prod_{j=1}^{k} H(z)_j \cdot H'(z)_j = \prod_{j=1}^{k} \frac{\rho_j^2 - 2\rho_j \cos\theta_j z^{-1} + z^{-2}}{1 - 2\rho_j \cos\theta_j z^{-1} + \rho_j^2 z^{-2}}.$$

All that was needed for this method was a set of band bounds and a predetermined β-value. The result is a filter with a transfer function with structure that we understand and an order we have some influence over. We also know all the poles and zeros. These features make this method very useful.

We can plot the group delay $\tau(\omega)$, where $\omega \in [-\pi, \pi]$, of each first-order all-pass section by using the following relation

$$\tau(\omega) = \frac{1-\rho^2}{1+\rho^2-2\rho\cos(\omega-\theta)}.$$

These sections all integrate to 2π and the summation of these sections results in the group delay of the dispersion filter [19].

Example 3.4.1. We create a filter of order 10 by using 5 stable biquad sections. The sampling rate in this example is 14 kHz, so the Nyquist limit is at 7 kHz. The filters group delay increases as a signal frequency approaches this Nyquist limit, meaning the amplitude envelope will be delayed more at higher frequencies.

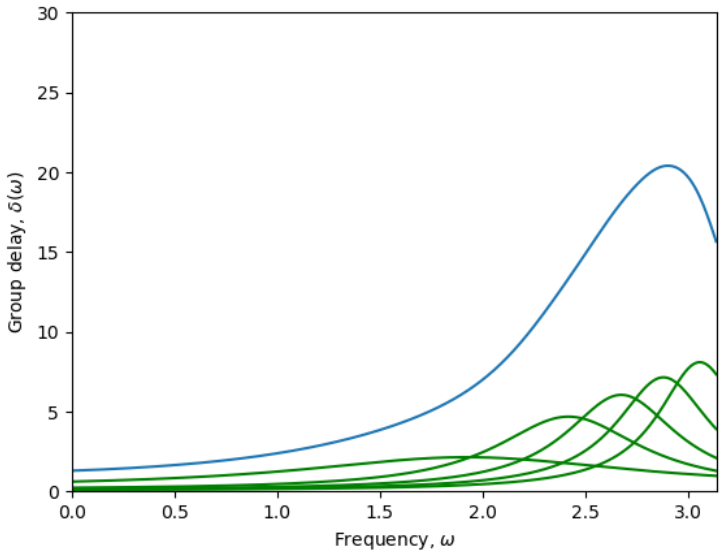

Figure 3: The group delay of the dispersion filter in example (3.4.1) consisting of five stable biquad sections. The group delay of each individual all-pass section is plotted in green.

△

Example 3.4.2. In this example we design a filter using twenty biquad sections, creating a model of a much larger order, which is fourty. This allows for a much larger group delay at higher frequencies, as the group delay needs to integrate to a larger multiple of 2π. The sampling rate is again 14 kHz.

△

In the next sections we will perform order reductions for these two examples and compare their results.

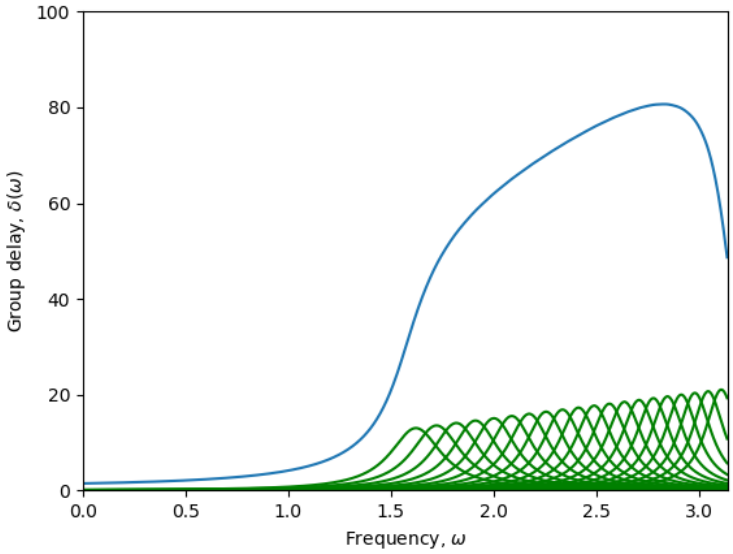

Figure 4: The group delay of the dispersion filter in example (3.4.2) consisting of twenty stable biquad sections. The group delay of each individual all-pass section is plotted in green.

4 Experiments

In this section, we will discuss several examples of audio filters by applying both balanced and modal truncation and comparing their results in terms of possible errors. All these filters will be stable, as the poles used in the transfer function all have negative real parts. We will see if there are general properties that are found in all reductions. We would also like to see if one method constantly produces better results, or if both methods have their specific uses. We will examine models of different orders, and we would like to study the relative size of the reductions that can be made while only having small errors in the approximation, so when the error bounds have are close to zero.

Firstly, we will continue example (3.4.1) and then give another example. We can plot the error bounds of the possible reductions for example (3.4.1) in the same way as we did for (3.3.1). The results are found in figures (5) and (6).

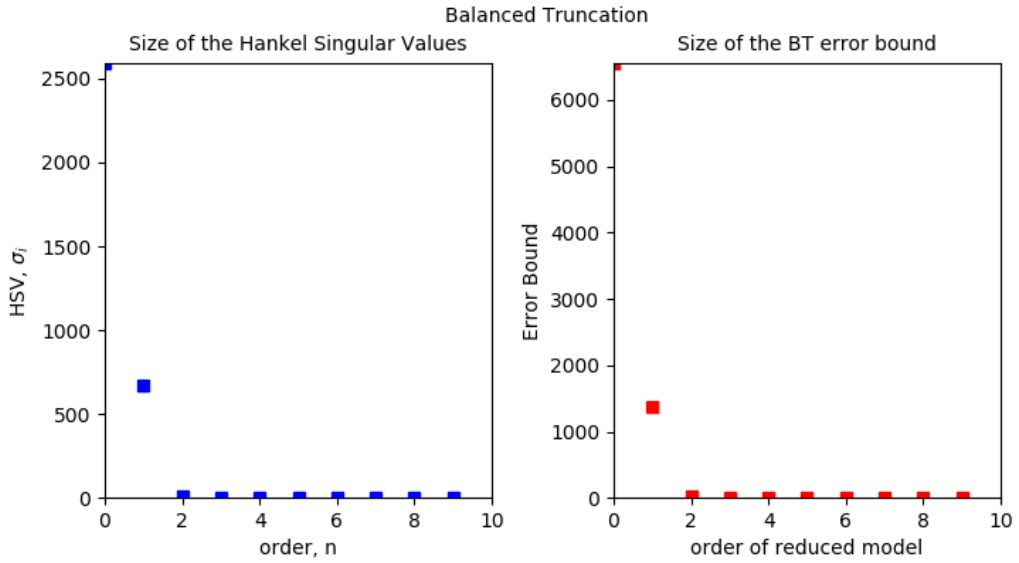

Figure 5: The size of the Hankel singular values of example (3.4.1) and its corresponding error bound at different reductions under balanced truncation.

The following is a similar case.

Example 4.0.1. Here, we use an order 10 filter with a transfer function consisting of five biquad sections as described in section (3.4). It models a group delay as plotted in figure (7). This filter is similar in terms of order to example (3.4.1), but models a more detailed group delay.
The results of using both methods to truncate the filter are found in figures (8) and (9).

△

There are a few things that stand out when analysing these error bounds of examples (3.4.1) and (4.0.1). Firstly, the possible error in a reduced system becomes larger in all cases, when making a larger reduction. This is expected behaviour of course. It is clear that the error becomes enormous when reducing to a system of order one or zero for both examples, but order reductions smaller than 40% seem to give relatively small errors. An exception seems to be the case of modal truncation applied to example (3.4.1), where there is a large possible error for all reductions.
Because of the extreme increase in errors that comes with greater reductions, it is hard to see at what point the errors become almost zero. We give a better view for these values of the balanced truncation method for example (4.0.1) in figure (10). Example (3.4.1) gives similar results, the plot is ommitted here.

We can visualize the accuracy of reductions in another way, by showing the impulse response of the filter at different reductions. This is done for example (3.4.1) in figure (11) for the original filter and three reductions

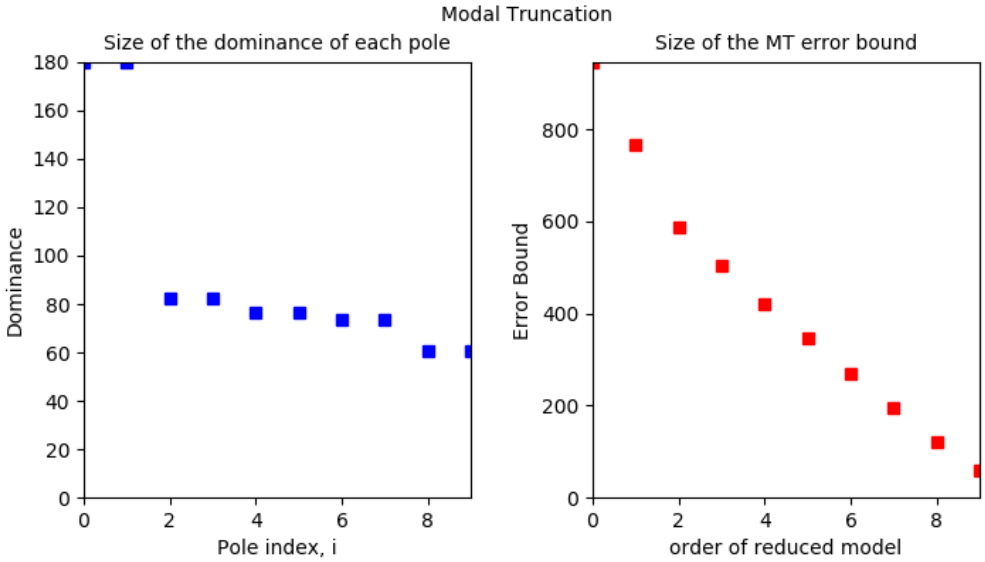

Figure 6: The dominance of the poles of the transfer function of example (3.4.1) and its corresponding error bound at different reductions under modal truncation.

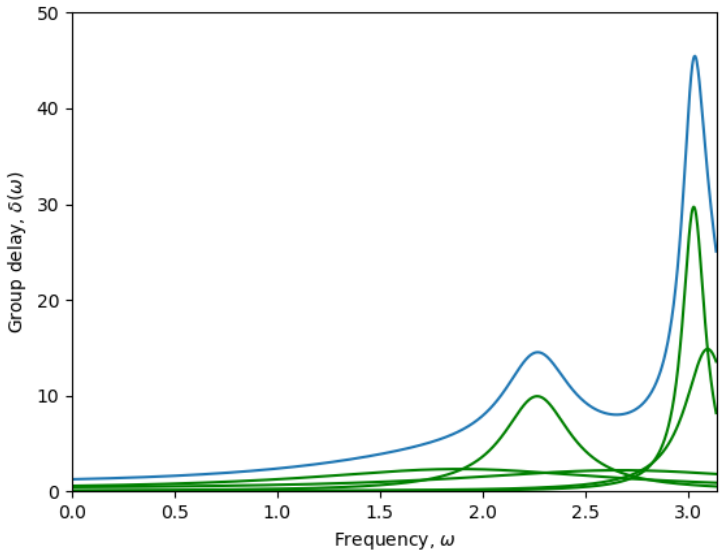

Figure 7: A plot (blue line) of the group delay imposed by the filter of example (4.0.1). This filter has a transfer function of 5 cascaded biquad sections, each applying a group delay that is plotted by a green line.

using balanced truncation. The impulse response fully characterizes LTI filters[5], and will give us some idea of the succes of a reduction.

There exists only a small noticable difference between the impulse response of the original filter and of the filter of order 9. When reduced to order 7, the filter still approximates the impulse response quite well during the first 15 samples, but most detail is lost between samples 15 and 40. The order-3 filter has an impulse response that is clearly very different from the original filter. These are results that can be expected considering the error bounds in figure (5). Applying balanced truncation to example (4.0.1) produces very similar results when plotting the impulse responses, these figures are ommitted.

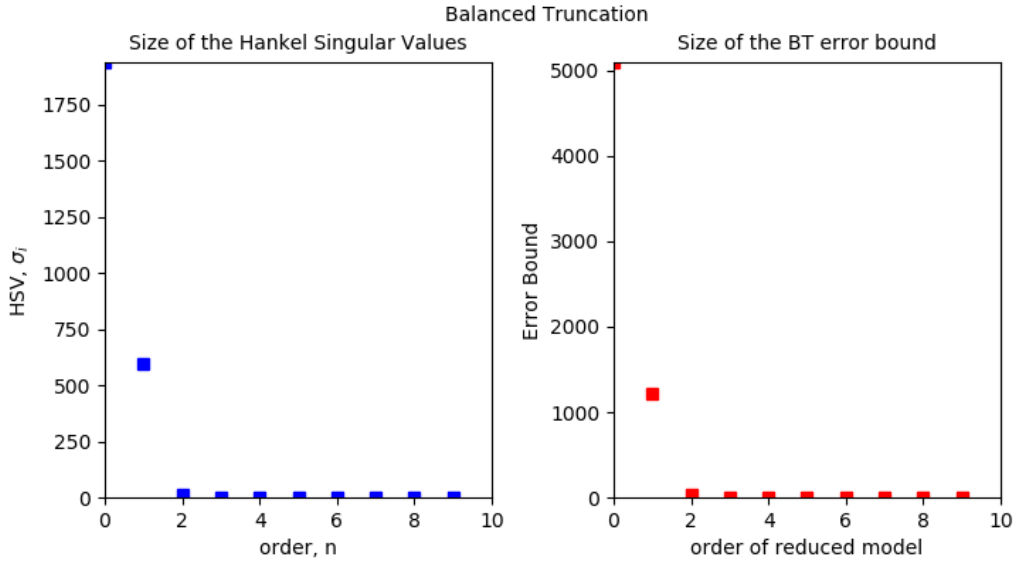

Figure 8: The size of the Hankel singular values of example (4.0.1) and its corresponding error bound at different reductions under balanced truncation.

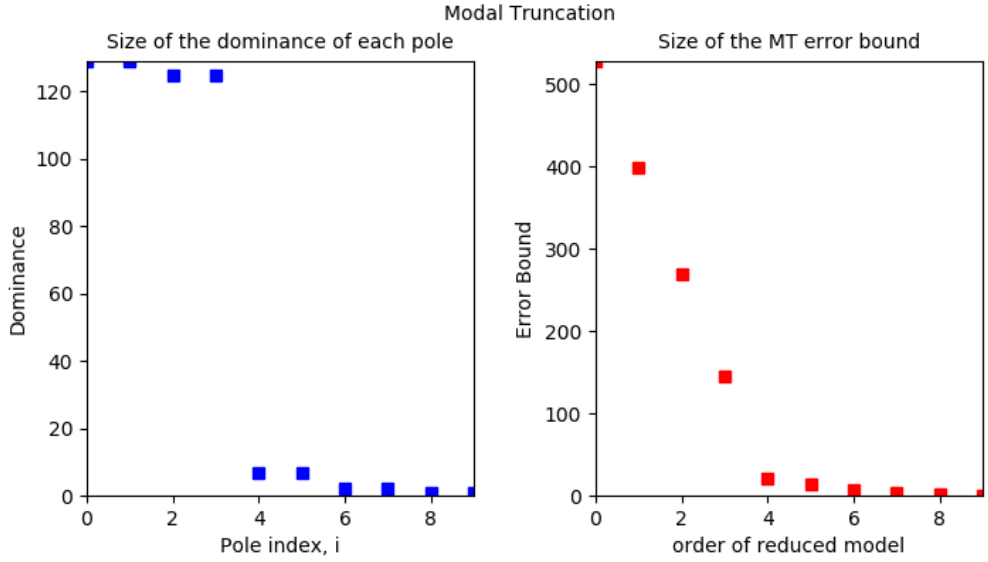

Figure 9: The dominance of the poles of the transfer function of example (4.0.1) and its corresponding error bound at different reductions under modal truncation.

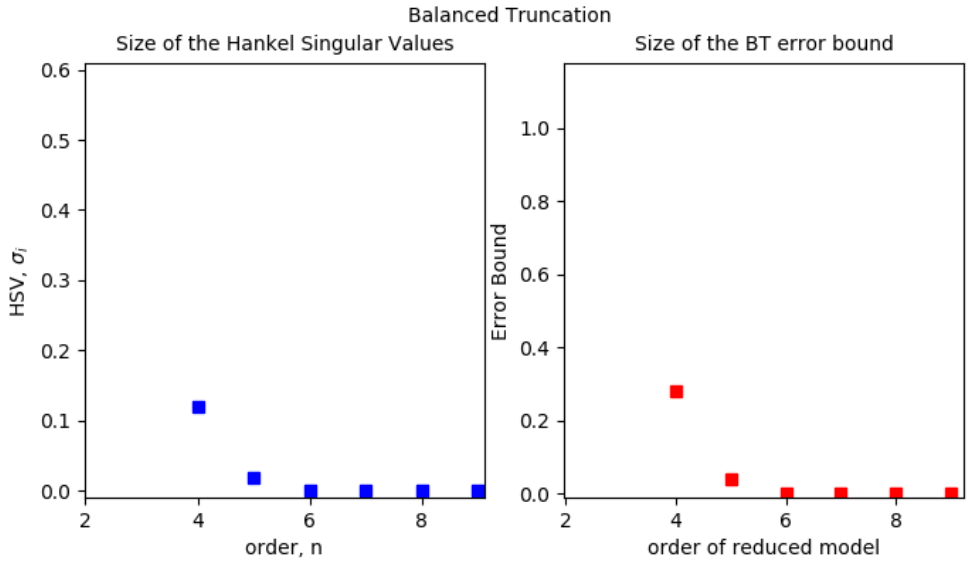

Figure 10: A close-up view of the Hankel singular values and error bound of example (4.0.1). Hankel singular values corresponding to lower orders than shown are very large.

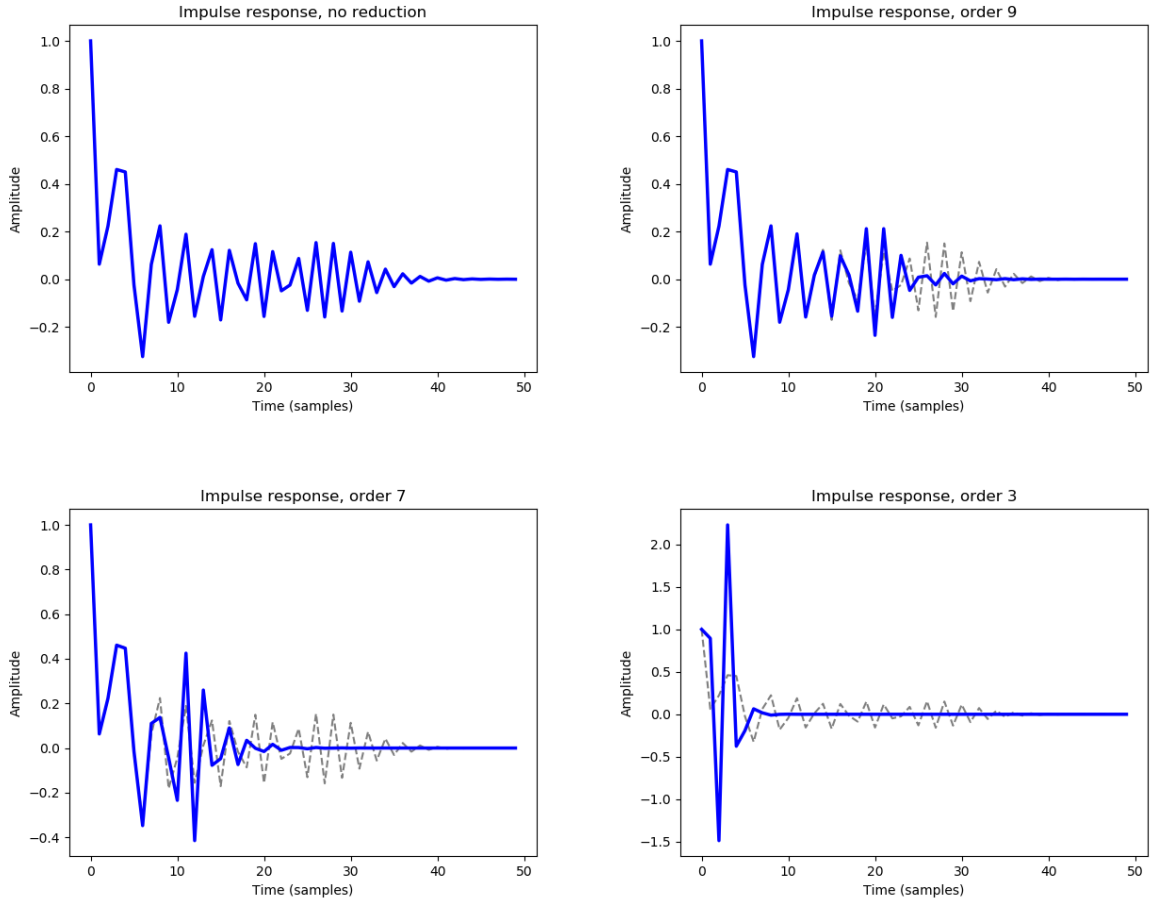

Figure 11: The impulse response of filter (4.0.1) and the approximations at three different order. The impulse response of the original filter is plotted in each figure with a dotted grey line.

We will now continue example (3.4.2) by showing the error bound data for both methods in figures (12) and (13). We only show where the error bound approaches zero here, for all cases the values becomes extremely large at greater reductions. We will also discuss a new example.

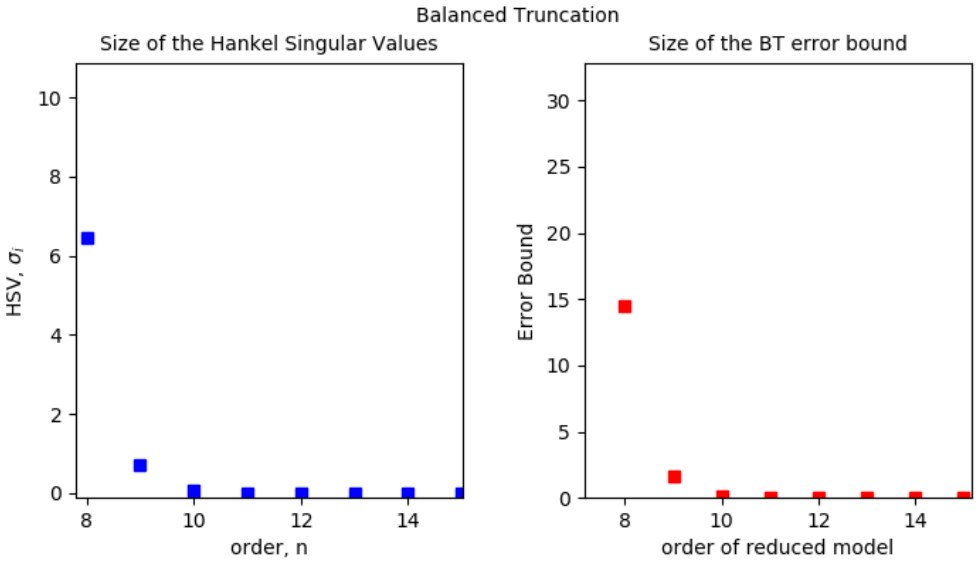

Figure 12: The size of the Hankel singular values of example (3.4.2) and its corresponding error bound at different reductions under balanced truncation.

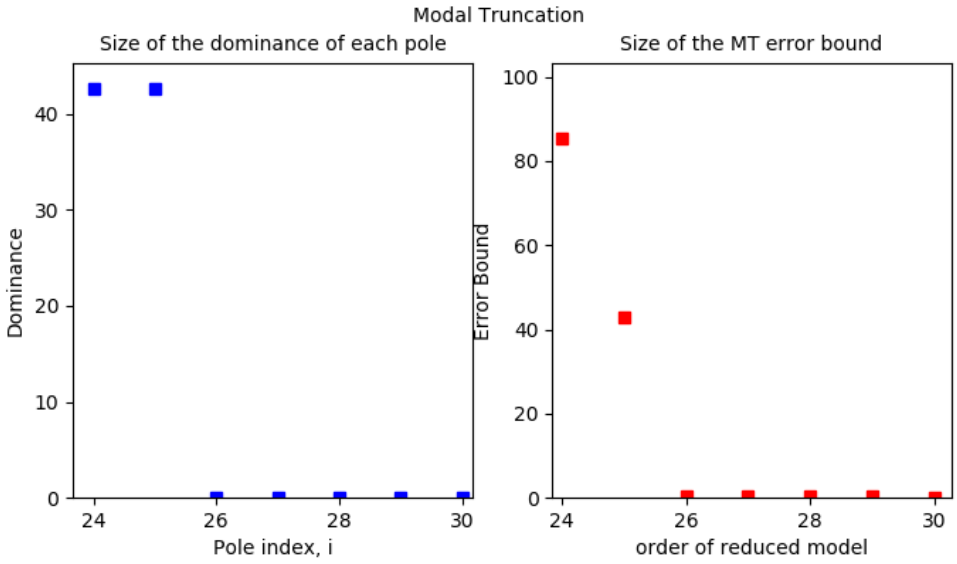

Figure 13: The dominance of the poles of the transfer function of example (3.4.2) and its corresponding error bound at different reductions under modal truncation.

Example 4.0.2. Another filter that we will reduce is also made up of twenty cascaded biquad sections, so it has order 40. The group delay it imposes is shown in figure (14). The filter has a more detailed group delay, but the same order as filter (3.4.2).

△

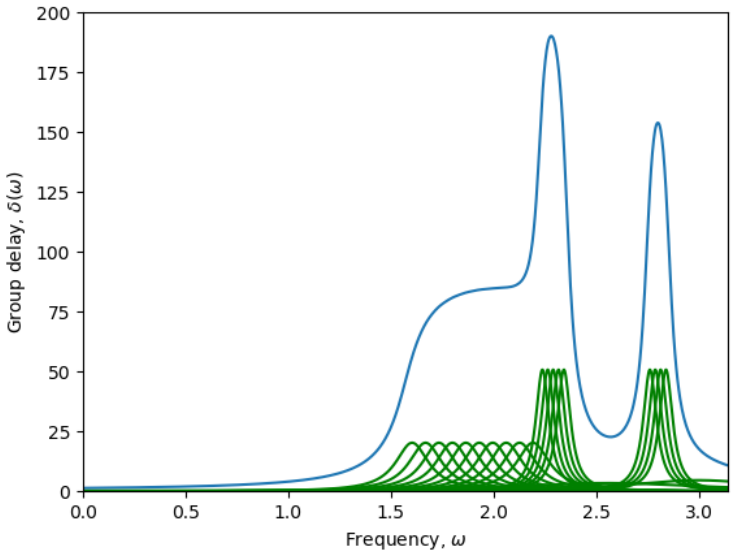

Figure 14: A plot (blue line) of the group delay imposed by the filter of example (4.0.2). This filter has a tranfer function of 20 cascaded biquad sections, each applying a group delay that is plotted by a green line.

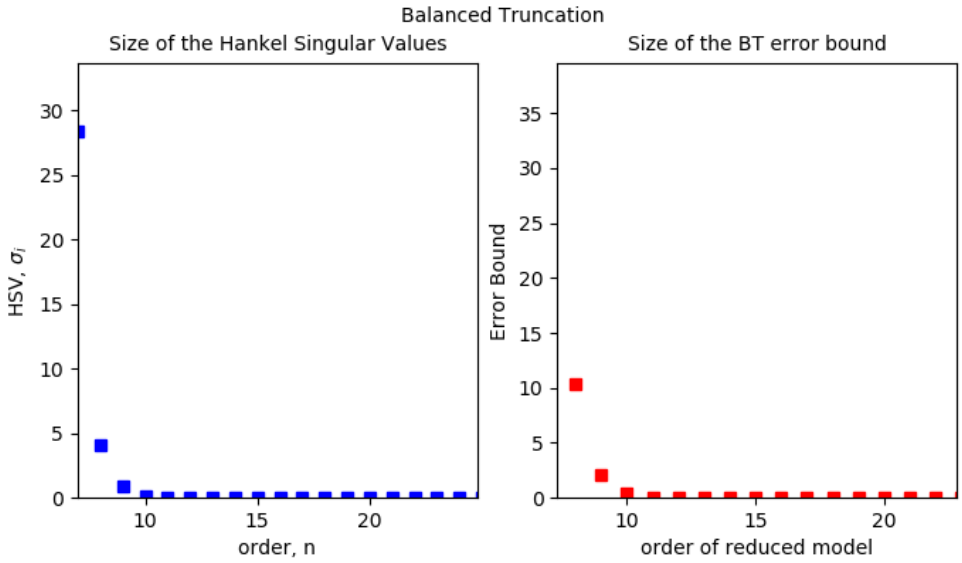

Figure 15: The size of the Hankel singular values of example (4.0.2) and its corresponding error bound at different reductions under balanced truncation.

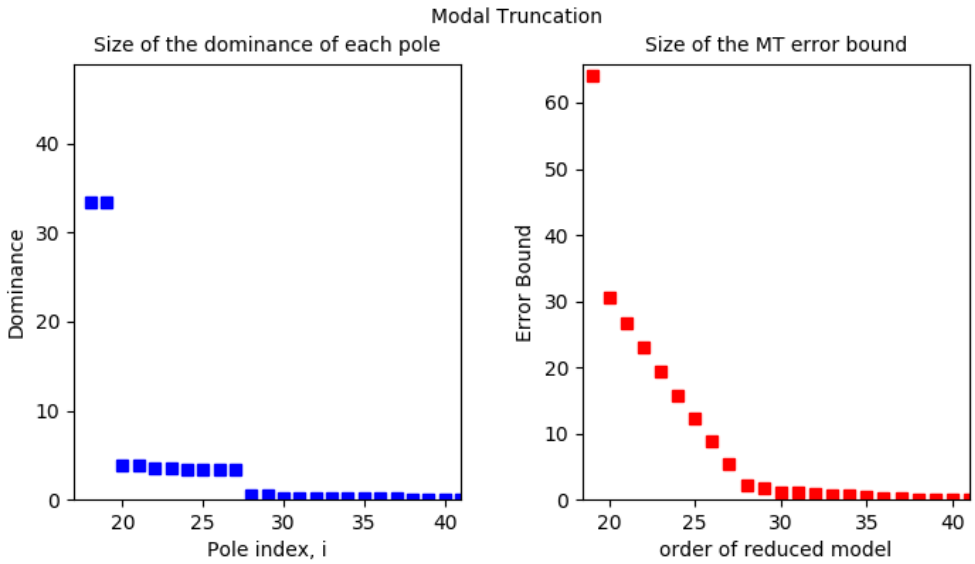

Figure 16: The dominance of the poles of the transfer function of example (4.0.2) and its corresponding error bound at different reductions under modal truncation.

These examples of filters of order 40 show, like previous examples, that the Hankel Singular Values and dominance of the poles corresponding to higher orders approach zero very rapidly at some point. The error bound that belongs to reductions to these orders approach zero as well. We see that for these high-order filters balanced truncation produces great results, where a reduction to order 11 has only a small error. This is an excellent result. In [20], Mackenzie, Kale and Cain use balanced truncation to reduce a very high-order model and find a possible reduction of a similar size. Modal truncation achieves less impressive results in these cases. In example (3.4.2), a reduction to order 26 seems acceptable, and for example (4.0.2) we can only reduce the order to the 30-35 range. This is still a moderately positive outcome, but modal truncation is significantly less effective here than balanced truncation.

When evaluating all examples, we find that almost all filters can be reduced in order by using both methods. Balanced truncation produces the best approximations at greater reductions. These reductions are in several cases greater than 50%, while only keeping a very small error bound. Modal truncation allows the order to be reduced a little, but never more than 30%, without generating large errors in the approximation. We have examined filters of different orders, and for all orders we can do about the same relative size of reduction. Most importantly, we have shown that both methods can be used to reduce the order of digital audio filters and produce viable results.

5 Conclusion

We gave a short introduction to model order reduction and both balanced and modal truncation have been examined in detail. The necessary theory has been covered and the methods are discussed in terms of the properties they provide like stability preservation and computable error bounds. We have examined how to apply these methods to the real-world application of digital audio filters and presented the results of different examples.

Both truncation methods allowed for reductions of several orders, sometimes more than 50%, while preserving the input-output behaviour of the original model. For the filters examined in this thesis, we can conclude that balanced truncation seems to allow for larger order reductions while keeping the possible error close to zero. This was the case for filters of both moderate and high orders. We used filters that generate a certain group delay, that were all constructed via the same method. If the comparison between balanced and modal truncation were to be researched further, another method to construct these filters could be used, to see if the methods provide a smaller or larger advantage. Next, different types of filters might be examined, like various types of low-pass or high-pass filters. These are filters with other characteristics, and the methods might accomplish different results. It would also be possible to examine the effects of projection based MOR techniques on these filters. As different techniques are specialized to certain systems to a varying degree, there might be significant differences in the sizes of reductions that are possible.

To conclude, we have given a good overview of two truncation based MOR techniques and their viability in reducing the order of digital audio filters. We have found that for a specific type of filter the methods can be used to greatly reduce the order of the system, while assuring that its output is similar to the original.

References

[1] B. Moore. "Principal component analysis in linear systems: Controllability, observability, and model reduction". In: *IEEE Transactions on Automatic Control* 26.1 (1981). DOI: 10.1109/TAC.1981.1102568.

[2] K. Glover. "All optimal Hankel-norm approximations of linear multivariable systems and their L-infinity error bounds". In: *International Journal of Control* 39.6 (1984). DOI: 10.1080/00207178408933239.

[3] A.C. Antoulas et al. *Model order reduction : methods, concepts and properties*. Vol. 1507. CASA-report. Technische Universiteit Eindhoven, 2015.

[4] W.H.A. Schilders. "Introduction to Model Order Reduction". In: *Model Order Reduction: Theory, Research Aspects and Applications*. Mathematics in Industry, vol. 13. Springer, Jan. 2008. ISBN: 978-3-540-78840-9. DOI: 10.1007/978-3-540-78841-6_1.

[5] J.O. Smith. *Introduction to Digital Filters with Audio Applications*. Online book, accessed April-May 2020. http://ccrma.stanford.edu/ jos/filters/, 2007.

[6] E. Sontag. *Mathematical Control Theory: Deterministic Finite-Dimensional Systems*. Vol. 6. Jan. 1998, p. 11. ISBN: 978-1-4612-6825-3. DOI: 10.1007/978-1-4612-0577-7.

[7] J. Rommes. "Modal Approximation and Computation of Dominant Poles". In: *Model Order Reduction: Theory, Research Aspects and Applications*. Mathematics in Industry, vol. 13. Springer, Jan. 2008. ISBN: 978-3-540-78840-9. DOI: 10.1007/978-3-540-78841-6_1.

[8] R.A. Horn and C.R. Johnson. *Matrix Analysis (Second ed.)* Cambridge University Press, 2013, pp. 441–442. ISBN: 978-0-521-83940-2.

[9] B.C. Hall. *Lie Groups, Lie Algebras, and Representations: An Elementary Introduction*. Graduate Texts in Mathematics. Springer International Publishing, 2015, pp. 31–35. ISBN: 978-3-319-13466-6. DOI: 10.1007/978-3-319-13467-3.

[10] A.C. Antoulas and D. Sorensen. "Approximation of Large-Scale Dynamical Systems: An Overview". In: *International Journal of Applied Mathematics and Computer Science* 11 (Oct. 2001). DOI: 10.1016/S1474-6670(17)31584-7.

[11] M. Green and D. Limebeer. *Linear robust control*. Prentice-Hall information and system sciences series. Prentice Hall, Inc., 1995, pp. 314–318. ISBN: 978-0-486-48836-3. DOI: 10.1016/0967-0661(95)90063-2.

[12] J. Rommes and N. Martins. "Efficient Computation of Transfer Function Dominant Poles Using Subspace Acceleration". In: *Power Systems, IEEE Transactions on* 21 (Sept. 2006), pp. 1218–1226. DOI: 10.1109/TPWRS.2006.876671.

[13] J. Rommes and N. Martins. "Efficient Computation of Multivariable Transfer Function Dominant Poles Using Subspace Acceleration". In: *Power Systems, IEEE Transactions on* 21 (Dec. 2006), pp. 1471–1483. DOI: 10.1109/TPWRS.2006.881154.

[14] P. Benner and E.S. Quintana-Orti. "Solving Stable Generalized Lyapunov Equations with the Matrix Sign Function". In: *Numerical Algorithms* 20 (Mar. 1999), pp. 75–100. DOI: 10.1023/A:1019191431273.

[15] N.S. Ellner and E.L. Wachspress. "Alternating Direction Implicit Iteration for Systems with Complex Spectra." In: *SIAM Journal on Numerical Analysis 28* 3 (1991), pp. 859–870. DOI: 10.1137/0728045.

[16] J. Phillips and L. Silveira. "Poor man's TBR: a simple model reduction scheme." In: 2 (2004), pp. 938–943. DOI: 10.1109/DATE.2004.1269012.

[17] C. Chen. *Linear System Theory and Design*. Third edition. Oxford University Press, 1999, pp. 135–137. ISBN: 0-19-511777-8.

[18] G. Gu. "All optimal Hankel-norm approximations and their L-infinity error bounds in discrete-time". In: *International Journal of Control* 78:6 (2005). DOI: 10.1080/00207170500110988.

[19] J.S. Abel and J.O. Smith. "Robust Design of Very High-Order Allpass Dispersion Filters". In: Proc. of the 9th Int. Conference on Digital Audio Effects (DAFx-06) (2006). September 18-20.

[20] J. Mackenzie, I. Kale, and G.D. Cain. "Applying Balanced Model Truncation To Sound Analysis/synthesis Models". In: (Nov. 1995).