Thesis for the Degree of Licentiate of Engineering

Model Order Reduction in FEMLAB by Dual Rational Arnoldi

K. Henrik A. Olsson

CHALMERS | GÖTEBORG UNIVERSITY

Department of Mathematics
Chalmers University of Technology and Göteborg University
SE-412 96 Göteborg, Sweden
Göteborg, August 2002

Model Order Reduction in FEMLAB by
Dual Rational Arnoldi
K. HENRIK A. OLSSON

ISSN 0347-2809/NO 2002:57
Department of Mathematics
Chalmers University of Technology and Göteborg University
SE-412 96 Göteborg
Sweden
Telephone +46 (0)31-772 1000

This is a thesis of the ECMI (European Consortium for Mathematics
in Industry) post-graduate program in Industrial Mathematics at
Chalmers University of Technology.

The work was supported by COMSOL AB.

Matematiskt centrum
Göteborg, Sweden 2002

Abstract

A dual rational Arnoldi method for model order reduction has been implemented. It is intended for use in the finite element software FEMLAB®. The algorithm is designed for generalised state space systems with large sparse matrices as well as multiple inputs and multiple outputs. Basis matrices for unions of Krylov subspaces are constructed to ensure a moment matching property of the reduced order model, and it is shown how to deflate redundant vectors during this construction. The choice of shift points and the use of a sparse LU-factorisation or an iterative method for solving linear systems are discussed.

The reduced order approximation of a stable system may be unstable. A method for stabilising such a model by removing its unstable modes has been implemented, and other stabilisation ideas in the literature are mentioned.

Two FEMLAB models, one of a convection problem and the other of a steel floor loaded by two vibrating machines, are described and used for testing the algorithms. The reduced order models are compared with the full order models, in terms of both transfer functions and results from time-steppings.

Sammanfattning

En dual rationell Arnoldimetod för modellreduktion har implementerats. Den är avsedd för användning i finita elementprogramvaran FEMLAB®. Algoritmen är anpassad för generaliserade tillståndsrumssystem med stora glesa matriser och flera insignaler och utsignaler. Basmatriser för unioner av Krylovrum konstrueras för att garantera att vissa moment har samma värden för det reducerade systemet som för det ursprungliga, och det visas hur överflödiga vektorer tas bort under denna konstruktion. Valet av skiftpunkter och användningen av en gles LU-faktorisering eller en iterativ metod för lösning av linjära ekvationssystem diskuteras.

Den reducerade modellen kan vara instabil även om den ursprungliga modellen är stabil. En metod för stabilisering genom borttagande av de instabila moderna har implementerats, och andra kända stabiliseringsidéer nämns.

Två FEMLAB-modeller, en av ett konvektionsproblem och den andra av ett stålgolv belastat av två vibrerande maskiner, beskrivs och används för att testa algoritmerna. De reducerade modellerna jämförs med de ursprungliga modellerna med avseende på både överföringsfunktioner och resultat av tidsstegningar.

Keywords: Model order reduction, dual rational Arnoldi algorithm, moment matching, rational Krylov

AMS 2000 subject classification: 65F15, 93A30, 65M60, 65F50, 65F30

Acknowledgements

I thank:

my supervisor Axel Ruhe for supervising my work and for reading this manuscript at several stages,

COMSOL for financial support of this project, and Lars Langemyr, Daniel Bertilsson and everybody else at COMSOL who have helped me during my time there,

CESAME, Université catholique de Louvain, and especially Paul Van Dooren, for great hospitality during my four month visit,

Daniel Skoogh for participating in the presentation of this work,

Greger Cronquist for being a helpful LaTeXguru, and also for reading this manuscript,

August Strindberg, Charles Mingus and Elvis Costello for inspiring words and music,

my family and friends for everything.

About ECMI

This licentiate thesis concludes a five semester ECMI (European Consortium for Mathematics in Industry) postgraduate programme. Of this time, about three fifths is spent taking courses in mainly applied mathematics, one fifth is spent teaching, and the rest is spent on the project presented in the thesis. At some time during the programme, a period of about four months should be spent at a foreign university. The thesis project should be based on an industrial problem.

About FEMLAB

FEMLAB® is a software for solving partial differential equations using the finite element method. It is produced by COMSOL AB and used with MATLAB®. Information about FEMLAB and COMSOL can be found on `http://www.femlab.com`.

Contents

Chapter 1

Introduction

1.1 General Introduction

In mathematical modelling we often face an accuracy versus complexity trade-off, in situations where a model that is in some sense larger or more difficult to handle gives better results compared with a basic model. Two possible ways to approach this are to either choose the most simple model that gives a required accuracy or choose the most complex model that can be dealt with and estimate the accuracy afterwards. Of these, the first is usually more sound, but in some cases the second may be the only option.

For some models, the size and complexity are described by a quantity known as the order. For example, the order of a model can be the size of a matrix involved in the model. In some modelling contexts, the order follows from the choice of one or more parameter values. Then we could pick these values such that the model becomes sufficiently small. The alternative that we consider is to use other parameter values and generate a larger model, and then reduce the size of this model through a method for model order reduction. Hopefully, for a given maximal order this gives a model that is more accurate than the one obtained for the corresponding parameter values.

The aim of our work presented in this thesis is to study and implement model order reduction in the finite element software FEMLAB®. The order of a finite element model is usually the number of degrees of freedom in the finite element approximation, and a typical parameter value that affects the order is the maximum element size in the mesh.

1.2 Linear Time-Invariant Dynamical System from Finite Element Discretisation in FEMLAB

The systems we study come from discretisations of partial differential equations using the finite element method. These partial differential equations are defined on domains in one or more space dimensions, and have solutions that belong to infinite dimensional function spaces. For time-dependent problems, we use finite elements for the space variables only; other methods are used for the time-stepping. The finite element discretisation consists of a subdivision of the space domain into a mesh of smaller objects, and the construction of a finite dimensional vector space related to this mesh where an approximate solution is sought. In this way, we can use a computer for solving problems to which we cannot derive an analytical solution. In general, a finer discretisation gives a larger problem with a solution closer to that of the original problem. The processor speed and memory size of the computer set the limit for how large we can make these problems, and especially for problems in three space dimensions, we find that if we make the discretisation as fine as we would like to, we cannot solve the resulting equations. We see an example of a finite element mesh of a two-dimensional space domain in Figure 1.1. For details on the finite element method, see,

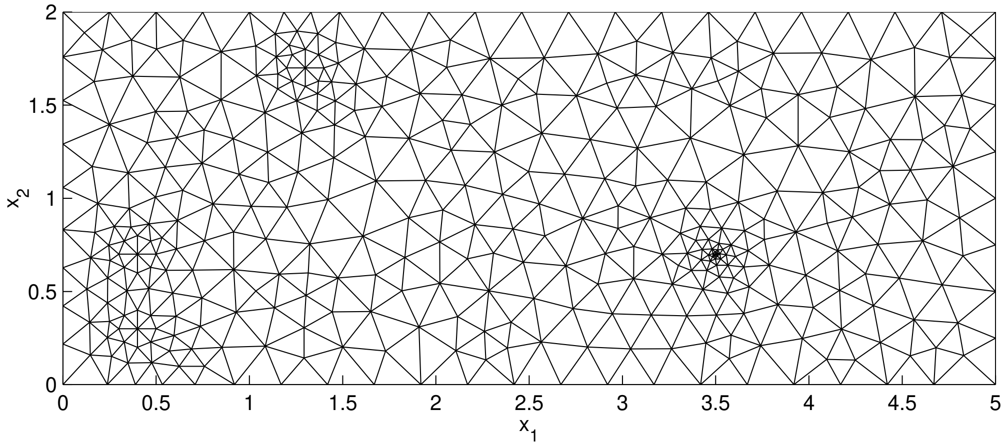


Figure 1.1: A finite element mesh generated with FEMLAB. This mesh is used for the vibrating machines model in Section 4.2.

for example, the textbook by Eriksson et al. [12].

We study finite element models in FEMLAB. Such models can be linear or nonlinear in the variables that are solved for, and time-dependent in various ways. For details on how the resulting systems are formulated and solved, and also for more information on FEMLAB, we refer to the reference manual [9]. The models we consider are time-dependent and formulated with some input signals and some output signals, and the main interest is in computing the output signals for given input signals. If such a model is not linear, we linearise it at, for example, a steady state solution. Thus, it can be written as the linear time-invariant dynamical system

$$\begin{aligned} E\dot{x} &= Ax + Bu, \\ y &= Cx + Du, \end{aligned} \tag{1.1}$$

where x is a vector of the unknown function values at the mesh points, u is a time-dependent input function and y is the output function. The dot over x denotes differentiation with respect to time. This system is usually called a *state space system* if the matrix E is equal to the identity matrix I, and a *generalised state space system* or *descriptor system* otherwise. We use state space system as a generic name for systems of the form in (1.1), and the word standard when we want to emphasise that E has to be the identity and the word generalised when we want to emphasise that E can be general. Some authors use the name descriptor system to indicate that E is singular.

In the state space system (1.1), the vector x is called the state and its components are called state variables. In FEMLAB, this system is obtained through a state space export using the function *femstate*. For examples of this kind of models, see Chapter 4. The system can also be created with the function *femsim* for export to the MathWorks product Simulink and use in simulations there. In that case, it is important that the system has a rather small number of state variables. The main reason for this is that Simulink does not support sparse matrices, so the matrices of the system (1.1) are stored as full matrices; for a large system this requires a lot of memory and makes simulations very time-consuming. One way to obtain a small model is to use a very coarse discretisation, but often this is not feasible since it leads to an unacceptably large error in the finite element solution. Another way is to start from a large state space system from a sufficiently fine discretisation, and try to reduce the size of the state space system before the simulation in time is made. Then, the interesting approach that we consider is to seek a reduced order model where a low dimensional model shows as much as possible of the dynamic behaviour of the original system.

1.3 Model Order Reduction

We consider the linear time-invariant system (1.1), where the descriptor matrix $E \in \mathbb{C}^{n\times n}$ and the system matrix $A \in \mathbb{C}^{n\times n}$ are large and sparse.

A matrix is called sparse if it contains so many zeros that it is advantageous to store and operate on only the nonzero entries. For matrices obtained from finite element discretisations of a differential operator, it is also often true that the number of nonzero entries per row is bounded by a small constant that does not depend on n; the reason for this is that refining the mesh increases n but does not give each mesh point a larger number of neighbouring mesh points.

The descriptor matrix E may be singular, but we assume that the matrix pencil (A, E) is regular, that is, that there are only a finite number of values $\lambda \in \mathbb{C}$ such that $A - \lambda E$ is singular. The system may have multiple inputs and multiple outputs, with an input matrix $B \in \mathbb{C}^{n\times m}$ and an output matrix $C \in \mathbb{C}^{p\times n}$, with $1 \leq m, p \ll n$. The sizes of these matrices is shown schematically in Figure 1.2. The term involving the matrix $D \in \mathbb{C}^{p\times m}$ is a direct

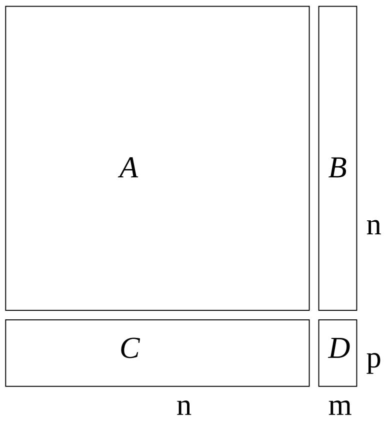


Figure 1.2: Schematic picture of state space matrices indicating the number of rows and columns of each matrix.

feed-through term from the input variables to the output variables, but since it does not affect the state variables it is not of interest in the order reduction. In the systems we study, D is a zero matrix, and if it were nonzero it could easily be eliminated from the system by a corresponding change in the output y.

Our aim in model order reduction is to obtain a system of small order r that is a good approximation of the original large system of order n. We note that the state vector x can be seen as the product Ix of a matrix whose columns are basis vectors and a vector of coefficients. This means that we use a basis consisting of n basis vectors to express the state vector. Do we really need all these basis vectors? The dynamics of the system may be such that most of it takes place in a subspace of low dimension. Then we could do fine with just a few basis vectors, and with a matrix V with these vectors as columns we could write $x = V\widehat{x}$, where $\widehat{x}$ are the time-dependent coefficients in the new basis. As an example, consider the simple problem

$$\begin{aligned} &\dot{x} = \alpha x,\ t > 0,\ \alpha \in \mathbb{C},\ x \in \mathbb{C}^n, \\ &x(0) = x_0, \end{aligned}$$

where the solution can be expressed as

$$x(t) = \frac{x_0}{\|x_0\|}\|x_0\|e^{\alpha t} \equiv v_1\widehat{x}_1$$

using only one basis vector. For more complicated problems, we need a way to generate the basis vectors and to decide how many such vectors are needed.

Note that the vectors in a basis are by definition linearly independent. Often they are also normalised to unit length and orthogonal.

To see how to obtain a reduced order system, we first consider the problem of solving the linear system of equations $Ax = b$, with $A \in \mathbb{C}^{n\times n}$. If we require that $x \in \operatorname{span}\{V\}$ where V has $r < n$ columns, we can write $x = V\widehat{x}$ as above and obtain the over-determined system $AV\widehat{x} = b$. It is quite likely that there is no exact solution to this system; in other words, for any approximate solution $\widehat{x}$ there will be a nonzero residual $\rho = b - AV\widehat{x}$. We can, however, make sure that this residual is orthogonal to a given subspace of dimension r by choosing a basis matrix Z for this subspace and solving the equation $Z^H\rho = 0$, where Z^H is the conjugate transpose of Z. Thus, we solve the equation $Z^HAV\widehat{x} = Z^Hb$ for $\widehat{x}$. This equation has a unique solution as long as the matrix Z^HAV is regular. We indicate the dimensions of the involved matrices in Figure 1.3. Note that in a Galerkin method, $Z = V$ is a matrix

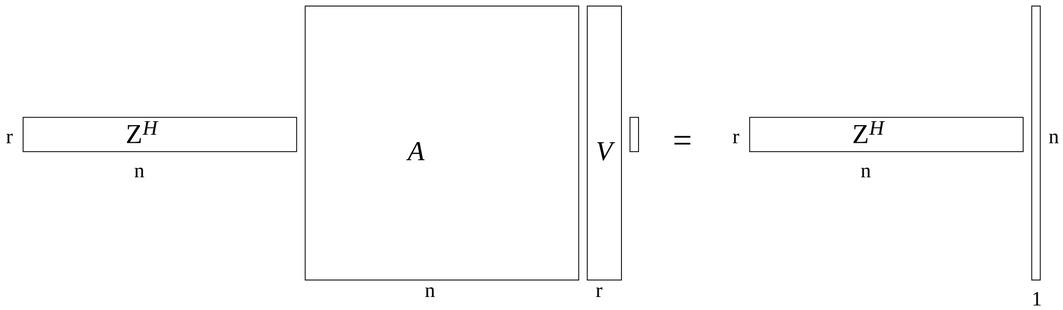


Figure 1.3: Schematic picture of a projected system of equations indicating the number of rows and columns of each matrix. The last vectors in the left and right hand sides are $\widehat{x}$ and b, respectively.

with orthonormal columns and the matrix $P = VV^H$ is an orthogonal projection. We will consider Petrov–Galerkin methods with oblique projections, where Z and V need not be equal but each has orthonormal columns. With $V \in \mathbb{C}^{n\times r}$ and a regular matrix $M \in \mathbb{C}^{r\times r}$, the columns of the matrices V and VM span the same spaces. For example, $x = V\widehat{x}$ if and only if $x = (VM)(M^{-1}\widehat{x})$. The choice of orthonormal columns in V is for numerical reasons when computations are made in finite precision on a computer.

In the same way as for the system of equations described above, we reduce the dimension of the linear time-invariant system (1.1) by requiring that the solution is in the subspace spanned by a matrix V and that the residual is orthogonal to the subspace spanned by a matrix Z. Thus, with $V, Z \in \mathbb{C}^{n\times r}$ and the full order system (1.1), we produce the reduced order system of order $r \ll n$

$$\begin{aligned} \widehat{E}\dot{\widehat{x}} &= \widehat{A}\widehat{x} + \widehat{B}u, \\ \widehat{y} &= \widehat{C}\widehat{x} + Du, \end{aligned} \tag{1.2}$$

where $\widehat{E} = Z^H EV$, $\widehat{A} = Z^H AV$, $\widehat{B} = Z^H B$, $\widehat{C} = CV$, and the approximate state vector is given by $x = V\widehat{x}$.

Now we turn to the question of how to choose the basis matrices V and Z in such a way that the reduced order system is a good approximation of the original system. There are, of course, various ways of measuring the difference between the two systems, and depending on what measure we choose, we might go for different reduction methods. The original system may have certain properties that are important to retain in the reduced order system; stability and passivity are typical examples. Then a system which has inherited these properties is better than one which has not. Often, we are most interested in the values of the output signals for given input signals, and we want these to be close for the full system (1.1) and the reduced system (1.2).

From a Laplace transformation of (1.1), with

$$X(s) \equiv \mathcal{L}\left(x(t)\right)(s) = \int_{0-}^{\infty} e^{-st}x(t)dt,$$

we get

$$X(s) = (sE - A)^{-1}\left\{Ex(0^-) + BU(s)\right\},$$

so with $D = 0$ and assuming zero initial conditions

$$Y(s) = C\left(sE - A\right)^{-1} BU(s) \equiv H(s)U(s), \tag{1.3}$$

where $H(s)$ is called the *transfer function*. The transfer function $\widehat{H}(s)$ for the reduced order system (1.2) is defined analogously. We note that the transfer function $H(s)$ is a $p\times m$ matrix, where the entry in position (j,k) is the scalar transfer function from input k to output j.

One way to find a good reduced order model would be to look for the one that minimises some norm of the error $H(\cdot) - \widehat{H}(\cdot)$. This is, for example, discussed in [30] for rather small control systems with $E = I$. Hankel norm approximation and balanced truncation are two methods with norm-wise error bounds for the resulting systems [2], but they are constructed for systems with $E = I$ and a dense matrix A and are not efficient for large, sparse systems. There is, however, work on efficient approximate versions of such methods on the way; see, for example, [36]. We use another criterion for the reduced order model that is better suited to the problems we study. Choosing a shift point σ such that $(A - \sigma E)$ is regular, we can rewrite $H(s)$ in the following way:

$$\begin{aligned} H(s) &= C\left(sE - A\right)^{-1} B = -C\left(A - \sigma E - (s-\sigma)E\right)^{-1} B \\ &= -C\left((A-\sigma E)(I - (s-\sigma)(A-\sigma E)^{-1}E)\right)^{-1} B \\ &= -C\left(I - (s-\sigma)(A-\sigma E)^{-1}E\right)^{-1}(A-\sigma E)^{-1}B. \end{aligned}$$

With a power series expansion we get

$$\begin{aligned} H(s) &= \sum_{j=0}^{\infty} -(s-\sigma)^j \, C\big\{ (A-\sigma E)^{-1} E \big\}^j (A-\sigma E)^{-1} B \\ &= \sum_{j=0}^{\infty} (s-\sigma)^j \, m_j(\sigma), \end{aligned}$$

where the coefficients

$$m_j(\sigma) \equiv -C\big\{ (A-\sigma E)^{-1} E \big\}^j (A-\sigma E)^{-1} B \tag{1.4}$$

are called moments at σ. We aim at constructing a rational interpolant, that is, a reduced order system such that some moments of $H(s)$ and $\widehat{H}(s)$ are equal in the expansions about a number of shift points σ_k. In a *Padé* approximation of a given order, a maximal number of moments are matched at the one shift point 0, and in a *partial realisation* moments are matched at ∞. If one finite nonzero shift σ is used we get a *shifted Padé* approximation, and in the general case with several shifts σ_k we get a *multipoint Padé* approximation.

We point out that we study causal systems, that is, systems where the output values at a certain time depend on past input values but not on future input values. Often the system is also stable, which means that it has a bounded output for all bounded inputs. Then, starting from zero initial conditions, a periodic input signal $u(t) = u_0 e^{i\omega t}$, with $u_0 \in \mathbb{R}^m$ and $i = \sqrt{-1}$, gives the periodic output signal $y(t) = H(i\omega) u_0 e^{i\omega t}$. This means that by matching the transfer functions so that $\widehat{H}(i\omega) = H(i\omega)$, the reduced order system gives the same output as the full order system when the input is a sine or cosine function of frequency ω.

We conclude this section by briefly mentioning the model order reduction method that was already used in FEMLAB when we started working on this thesis. In this method, called *mode reduction*, a basis matrix T is constructed from a given number of eigenvectors of the pencil (A, E) corresponding to the eigenvalues closest to zero, and optionally also the vectors in $-A^{-1}B$. The latter vectors are the static solutions for unit values on each of the inputs. The basis matrix is used for an orthogonal projection that gives a system of the form (1.2) with $V = Z = T$. This choice of eigenvectors should make the reduced order model a good approximation for slow changes in the state variables, and with the inclusion of the static solutions the stationary values of the state will be right. The eigenvectors are computed using the spectral transformation Arnoldi algorithm in the FEMLAB function *fleig*. Since eigenvalues close to zero are sought, the matrix $A - 0E$ is factorised in this algorithm. The important thing to note here is that this mode reduction involves one matrix factorisation unless an iterative method is used for solving systems with the shifted matrix. See the reference manual [9] for details on this mode reduction and the function *fleig*.

1.4 Krylov Subspace Methods

In early methods for moment matching, some moments of the full order system were computed and used for generating a reduced order system matching these moments. This was done either at zero or at infinity, that is, several shift points were not used. As we see in the definition of the moments (1.4), this explicit moment matching involves repeated multiplication with a fixed matrix. The numerical problems caused by this, and also how to avoid them through the use of a Lanczos method, are described by Gallivan, Grimme and Van Dooren [16] and also by Feldmann and Freund [13].

The first Krylov subspace methods for moment matching were designed for standard state space systems with only one input and one output, with matching at either zero or infinity [16] or at one arbitrary point [13]. Soon a Lanczos method for Padé approximation of systems with multiple inputs and outputs was given by Freund [14], and different methods were developed for systems with certain properties. In his thesis on Krylov projection methods [19], Grimme describes and compares several algorithms. Grimme's thesis and the references therein, combined with Freund's article [15] focused on circuit simulation, give a good overview of the field. There are several algorithms for model order reduction based on Krylov subspaces to choose from, designed for different state space systems and resulting in different reduced order models. We will now state a useful result and motivate our choice of a dual rational Arnoldi method, and then conclude this section by mentioning some related work on other methods.

Grimme [19] proves the following regarding the choice of the basis matrices V and Z for the reduced order system (1.2): If

$$\begin{aligned} &\bigcup_{k=1}^{K}\bigcup_{l=1}^{m}\mathcal{K}_{J_b(k,l)}\left\{\,(A-\sigma_k E)^{-1}E,(A-\sigma_k E)^{-1}B(:\,,l)\right\}\subseteq \operatorname{span}\left\{V\right\},\\ &\bigcup_{k=1}^{K}\bigcup_{l=1}^{p}\mathcal{K}_{J_c(k,l)}\left\{\,(A-\sigma_k E)^{-T}E^T,(A-\sigma_k E)^{-T}C(l,:\,)^T\right\}\subseteq \operatorname{span}\left\{Z\right\}, \end{aligned} \tag{1.5}$$

then the moments $m_j(\sigma_k)$ of the original and $\widehat{m}_j(\sigma_k)$ of the reduced order models satisfy

$$\begin{aligned} &-C(l_c,:\,)\left\{\,(A-\sigma_k E)^{-1}E\right\}^j(A-\sigma_k E)^{-1}B(:\,,l_b)=\\ &-\widehat{C}(l_c,:\,)\left\{\,\left(\widehat{A}-\sigma_k\widehat{E}\right)^{-1}\widehat{E}\right\}^j\left(\widehat{A}-\sigma_k\widehat{E}\right)^{-1}\widehat{B}(:\,,l_b) \end{aligned}$$

for $j=0,\ldots,J_b(k,l_b)+J_c(k,l_c)-1$; $k=1,\ldots,K$; $l_b=1,\ldots,m$; and $l_c=1,\ldots,p$.

Here $\mathcal{K}_k(A,v)\equiv\operatorname{span}\left\{v,Av,\ldots,A^{k-1}v\right\}$ denotes a Krylov subspace, J_b is a matrix where the element $J_b(k,l_b)$ in position (k,l_b) is the number of vectors to generate with the shift σ_k and the l_bth column of the input matrix B, and J_c is defined analogously to J_b. This is formulated for real systems; for complex systems it follows from the proof in [19] that the same result holds with the transpose T replaced by the conjugate transpose H.

Thus, we know that we obtain a desired moment matching as long as the inclusions (1.5) hold for our matrices V and Z, regardless of the method we use for creating these matrices. Our aim is to construct a robust algorithm for generating reduced order models that are as small as possible. Therefore, we use a dual rational Arnoldi algorithm for generating basis matrices with orthonormal columns and matching a maximal number of moments at a number of shift points. The method is called rational since it produces a rational interpolant, and dual because of the relation between the spaces spanned by V and Z. We describe this method and the algorithm in detail in Chapter 2. In an Arnoldi method where only the matrix V is constructed, and used for an orthogonal projection, the resulting reduced order model will in general match fewer moments. The exception is the symmetric case where $A=A^H$, $E=E^H$, and $B=C^H$. A Lanczos method could be used for generating biorthogonal matrices V and Z that satisfy the inclusions (1.5), but it would be more difficult to implement in a numerically stable way than the dual rational Arnoldi method.

There is a lot of related work on Krylov subspace methods for model order reduction, and we mention some of it that we find relevant here.

A rational Lanczos algorithm for single-input single-output systems is presented by Gallivan, Grimme and Van Dooren [17], and Skoogh [35] gives rational Krylov algorithms for the same kind of systems. Rational Krylov algorithms are also discussed in the talk by Ruhe and Skoogh [33]. The rational block Lanczos algorithm described by Nguyen and Li [29] and the rational block Arnoldi algorithm described by Elfadel and Ling [11] are related methods for systems with $m = p > 1$ inputs and outputs.

Grimme, Sorensen and Van Dooren [22] derive an implicitly restarted Lanczos method for standard state space systems with a single input and a single output. They use implicit restarting for stabilisation if the reduced order model is unstable. Another similar method is given by Jaimoukha and Kasenally [25]. Works aimed specifically at constructing stable or passive reduced order systems include the Lanczos method for stable partial Padé approximants by Bai and Freund [4] and the symmetric Lanczos method for passive systems by Bai, Feldmann and Freund [3].

Some other work on Krylov subspace methods that we find interesting is the derivation of error estimates for the Padé approximation via Lanczos done by Bai and Ye [5]; the presentation of a Lanczos method for multiple-input multiple-output systems, whose numbers of inputs and outputs need not be equal, by Aliaga et al. [1]; and the theoretical study of two-sided block Arnoldi and nonsymmetric Lanczos methods by Cullum and Zhang [10].

As examples of recent applications of the Krylov subspace methods we have considered, we would like to mention the use of a rational Arnoldi method by Chu and Lee [7] and the use of a dual rational Arnoldi method by Weile, Michielssen and Gallivan [38].

Finally, we find it appropriate to give a few references to related work on model order reduction that is not only focused on Krylov subspace methods. Antoulas and Sorensen [2] give an overview of model reduction of dynamical systems, and compare Krylov subspace methods with other methods such as balanced truncation and Hankel norm approximation. Stykel [37] presents balanced truncation for descriptor systems. Li [27], and Li and White [28], show the connection between a rational Krylov method and the alternate direction implicit algorithm for solving Lyapunov equations.

1.5 Notation

We use $\mathbb{R}$ and $\mathbb{C}$ to denote the fields of real and complex numbers, respectively, and $i = \sqrt{-1}$ is the imaginary unit. For a complex quantity x, $\operatorname{Re}\{x\}$ is its real part and $\operatorname{Im}\{x\}$ is its imaginary part, while $\overline{x}$ is the complex conjugate of x.

For parts of matrices, we use either MATLAB® style indexing or another notation, depending on which one is most easily read. Thus, given a matrix V, its kth column is denoted either $V(:\,,k)$ or v_k, and the submatrix consisting of its first k columns is denoted either $V(:\,,1\colon k)$ or V_k. Similarly, the element in row j and column k of V is usually denoted $V(j,k)$, but may also be denoted v_{jk} or V_{jk}. For rows we use MATLAB style indexing only, so $V(j,:\,)$ is the jth row of V. The matrix V^T is the transpose of V, while V^H is the conjugate transpose of V.

The matrix I is the identity matrix, of a size given by the context, and its kth column is denoted e_k.

We use $\operatorname{span}\{V_k\} = \operatorname{span}\{v_1, \ldots, v_k\}$ to denote the vector space spanned by the k columns of the matrix V_k, and similarly we define the Krylov subspace $\mathcal{K}_k(A, v) \equiv \operatorname{span}\{v, Av, \ldots, A^{k-1}v\}$.

We use the notation $\|\cdot\|$ for the Euclidean vector norm and the induced spectral matrix norm. Finally, we let $|V|$ be interpreted as the matrix with absolute values $|V(j,k)|$ as elements.

Chapter 2

Dual Rational Arnoldi

We use a dual rational Arnoldi algorithm for generating the basis matrices for the reduced order model, since we want to obtain a rational interpolant in a robust way. With this algorithm, we can handle multiple-input multiple-output systems, where the number of inputs and the number of outputs need not be the same. The resulting reduced order system matches a maximal number of moments, but it is not guaranteed to be stable even if the original system is.

We give a pseudo-code description of the algorithm here, and discuss a number of details in the rest of this chapter.

Algorithm 1 Dual rational Arnoldi

Input: State space matrices E, A, B and C; vector of shift points σ; vectors J_b and J_c where entry k is the number of vectors to generate with the kth shift and each starting vector from B and C, respectively; and deflation tolerance defltol.

Output: Orthogonal basis matrices V and Z.

function $[V, Z] = \text{dualRA}(E, A, B, C, \sigma, J_b, J_c, \text{defltol})$
initialise V and Z to empty matrices
for k from 1 to the number of shift points **do**
    LU-factorise $LU = A - \sigma_k E$
    $V = \text{Arnoldi}(L, U, E, B, V, J_b(k), \text{defltol})$
    $Z = \text{Arnoldi}(U^H, L^H, E^H, C^H, Z, J_c(k), \text{defltol})$
end for
if V and Z do not have the same number of columns **then**
    add more columns to the basis matrix with fewer columns
end if

We require that the input data to Algorithm 1 is consistent in the sense that the matrices V and Z get the same number of columns after the last step in the for loop if no deflation takes place. For example, assume that the state space system has one input and two outputs, which means that $B \in \mathbb{C}^{n\times 1}$ and $C \in \mathbb{C}^{2\times n}$. Then, if there is only one shift $\sigma = \sigma_1$ both J_b and J_c are scalars and they must satisfy $J_b = 2J_c$. In the general case where $\sigma \in \mathbb{C}^K$, $B \in \mathbb{C}^{n\times m}$ and $C \in \mathbb{C}^{p\times n}$ the condition is

$$m \sum_{k=1}^{K} J_b(k) = p \sum_{k=1}^{K} J_c(k). \tag{2.1}$$

This condition is modified in a rather special case; see the end of Section 2.3. If a different number of vectors are deflated in the constructions of V and Z in the for loop, we add more columns to the smaller matrix to get output matrices of the same size. We describe how this is done in Section 2.1.

The function Arnoldi is an Arnoldi algorithm with which we add vectors to a basis matrix; see Algorithm 2. It is designed to generate a basis for the desired union of Krylov subspaces. The index q on row 7 in Algorithm 2 is chosen such that v_q is the last column in V that was created with the present value of l. The deflation of $\widetilde{v}$ on row 11 in Algorithm 2 is done by not adding the vector to the basis and skipping the rest of the steps with the present value of l. We explain the choice of continuation vector, and the deflation, in Section 2.1.

Algorithm 2 Arnoldi

Input: Matrices L and U from factorisation of the operator, descriptor matrix E, right hand side matrix B, orthogonal basis matrix V (may be an empty matrix), number J of vectors to generate from each column in B, and deflation tolerance defltol.

Output: Orthogonal basis matrix V.

1: function $V = \text{Arnoldi}(L, U, E, B, V, J, \text{defltol})$
2: **for** j from 1 to J **do**
3:    **for** l from 1 to the number of columns in B **do**
4:       **if** $j = 1$ **then**
5:          $r = U^{-1}L^{-1}b_l$
6:       **else**
7:          $r = U^{-1}L^{-1}Ev_q$
8:       **end if**
9:       $\widetilde{v} = \text{GramSchmidt}(r, V)$
10:       **if** $\|\widetilde{v}\| < \text{defltol}\|r\|$ **then**
11:          deflate $\widetilde{v}$
12:       **else**
13:          $V = [V\ \widetilde{v}/\|\widetilde{v}\|]$
14:       **end if**
15:    **end for**
16: **end for**

The function GramSchmidt is an implementation of Gram–Schmidt orthogonalisation, with reorthogonalisation if necessary; see Algorithm 3. Note that the vector is orthogonalised but not normalised; the reason for this is that we use its norm in the deflation criterion in the Arnoldi algorithm.

The output from our dual rational Arnoldi algorithm consists of the basis matrices V and Z. We obtain the matrices of the reduced order state space system (1.2) through explicit multiplication—for example, $\widehat{A} = Z^H AV$—and these small matrices are in general full. This should be compared with Lanczos methods, where the reduced order matrices are built up from coefficients computed in the algorithm. Furthermore, because of the biorthogonality of the bases used in the Lanczos methods, these reduced order matrices have a certain sparsity structure.

Algorithm 3 Gram–Schmidt orthogonalisation

Input: Vector r and matrix V with orthonormal columns.
Output: Vector v orthogonal to $\operatorname{span}\{V\}$.

function $v = \mathrm{GramSchmidt}(r, V)$
$r_n = \|r\|$
$h_c = 0$
for i from 1 to 4 **do**
  $h_V = V^H r$
  $h_c = h_c + h_V$
  $r = r - V h_V$
  $r_{\text{norm}} = r_n$
  $r_n = \|r\|$
  if $r_n > 0.7 r_{\text{norm}}$ **then**
    break
  end if
end for
$v = r$

2.1 Continuation Vectors and Deflation

We construct basis matrices that satisfy the inclusions (1.5). For clarity, we describe the construction of the matrix V from the input matrix B; analogous statements hold true for the construction of Z from C. In the general case, B has m columns and we use K shifts σ_k, where both m and K are greater than one. This means that we must build a basis for a union of Krylov subspaces with different matrices and different starting vectors. For each shift σ_k, we need to solve systems with the matrix $A - \sigma_k E$. Therefore, we make the LU-factorisation $LU = A - \sigma_k E$, and to keep the memory requirements and CPU times as low as possible we have designed Algorithm 1 so that the shifts are used in turn and all operations with each shifted and inverted operator are done before moving on to the next shift; we give details on the LU-factorisation in Section 2.4. Starting with the shift σ_1, we generate $J_b(1)$ columns of V from each column in B, with Algorithm 2 acting as a regular vector-wise block Arnoldi algorithm with the starting block $(A - \sigma_1 E)^{-1} B$ and the matrix $(A - \sigma_1 E)^{-1} E$. Then we go to the next shift σ_2, and do a kind of restart by adding the new starting block $(A - \sigma_2 E)^{-1} B$ to our basis one vector at the time. With these new basis vectors and the matrix $(A - \sigma_2 E)^{-1} E$, we generate the part of V corresponding to the Krylov subspaces with shift σ_2. Each basis vector is orthogonalised against the previous vectors in V, so the vectors we operate on with the new matrix may in part be in the subspace corresponding to the old matrix. To see how this affects the constructed vectors, note that for the two different shifts σ_1 and σ_2 we get

$$(A - \sigma_2 E)^{-1} = (A - \sigma_2 E)^{-1}(A - \sigma_2 E + (\sigma_2 - \sigma_1)E)(A - \sigma_1 E)^{-1},$$

and thereby

$$(\sigma_2 - \sigma_1)(A - \sigma_2 E)^{-1} E (A - \sigma_1 E)^{-1} = (A - \sigma_2 E)^{-1} - (A - \sigma_1 E)^{-1},$$

which means that for each column b_l holds

$$(A - \sigma_2 E)^{-1} E (A - \sigma_1 E)^{-1} b_l \in \operatorname{span}\left\{(A - \sigma_2 E)^{-1} b_l,\ (A - \sigma_1 E)^{-1} b_l\right\}.$$

The equations above form the key step in the proof of the following lemma in [17], formulated for a system with a single input:

Lemma 1. *If σ_1 and σ_2 are two nonidentical shift points, then*

$$(A-\sigma_2 E)^{-1}E\left\{(A-\sigma_1 E)^{-1}E\right\}^{j-1}(A-\sigma_1 E)^{-1}b \in \\ \textit{span}\left\{(A-\sigma_2 E)^{-1}b\right\}\bigcup \mathcal{K}_j\left\{(A-\sigma_1 E)^{-1}E,(A-\sigma_1 E)^{-1}b\right\}$$

for any value of $j \geq 1$.

The point of this is that it shows that from such an operation on a previous vector we get a vector that lies in the union of the already constructed space and the new, desired space.

Note that we build the basis matrices one vector at the time and not by a block method, and that we want these matrices to have the same number of columns so that the reduced order system has quadratic matrices. We now explain this vector-wise construction and the concept of deflation through a detailed description of the construction of the basis matrix V.

To obtain an equality in the first inclusion in (1.5), we use Algorithms 1–3 to construct a matrix V with orthonormal columns such that

$$\operatorname{span}\{V\} = \operatorname{span}\left\{\left[R_1 \;\; M_1 R_1 \;\; \ldots \;\; M_1^{J_b(1)-1}R_1 \;\; R_2 \;\; M_2 R_2 \;\; \ldots \;\; M_K^{J_b(K)-1}R_K\right]\right\}, \tag{2.2}$$

where we use the notation $R_k \equiv (A-\sigma_k E)^{-1}B$ and $M_k \equiv (A-\sigma_k E)^{-1}E$ to save space. At each step of the construction of the matrix V, we add information from one of the columns of the matrix in the right hand side of equation (2.2), taken in order from left to right, so that $\operatorname{span}\{V(:,1)\} = \operatorname{span}\{R_1(:,1)\}$ and so on. The matrix in the right hand side has $m\sum_{k=1}^{K} J_b(k)$ columns, but these may be linearly dependent so that V has fewer columns. The removal of the columns that are linearly dependent on the previous ones is called deflation. In Algorithm 2 we deflate any candidate vector that has become zero after orthogonalisation. Note that, when V is constructed in this way, if $M_k^j R_k(:,l)$ was deflated then $M_k^{j+1}R_k(:,l)$ will be as well and need not be considered. This is why we can skip the rest of the steps in Algorithm 2 that correspond to the same matrix and starting vector as a deflated vector. Since we are working in finite precision on a computer, we also deflate the vector if its norm decreases too much in the orthogonalisation.

We now prove that we generate the desired bases. Theorem 1 is a generalisation of a similar theorem for a rational Lanczos algorithm for single-input single-output systems in [17]. The treatment of multiple starting vectors and deflation is inspired by [1].

Theorem 1. *If V and Z are the basis matrices constructed in Algorithm 1 with exact deflation, that is, defltol $= 0$, then after the end of the for loop*

$$\begin{aligned}
\operatorname{span}\{V\} &= \bigcup_{k=1}^{K}\bigcup_{l=1}^{m}\mathcal{K}_{J_b(k)}\{(A-\sigma_k E)^{-1}E,(A-\sigma_k E)^{-1}b_l\},\\
\operatorname{span}\{Z\} &= \bigcup_{k=1}^{K}\bigcup_{l=1}^{p}\mathcal{K}_{J_c(k)}\{(A-\sigma_k E)^{-H}E^H,(A-\sigma_k E)^{-H}C(l,:)^H\}.
\end{aligned}$$

Proof: To prove this, we use induction over an integer s to prove that the basis V_r after $s = s^*$ steps in Algorithms 1–2 with $k = k^*$, $1 \leq k^* \leq K$, $j = j^*$, $1 \leq j^* \leq J_b(k^*)$, $l = l^*$, $1 \leq l^* \leq m$, and

$$s^* = m\Big(\sum_{k=1}^{k^*-1} J_b(k)\Big) + l^* j^* + (m-l^*)(j^*-1),$$

satisfies

$$
\begin{aligned}
\operatorname{span}\{V_r\} &= \Big\{ \bigcup_{k=1}^{k^*-1} \bigcup_{l=1}^{m} \mathcal{K}_{J_b(k)}\big\{ (A-\sigma_k E)^{-1} E, (A-\sigma_k E)^{-1} b_l \big\} \Big\} \\
&\quad \bigcup \Big\{ \bigcup_{l=1}^{l^*} \mathcal{K}_{j^*}\big\{ (A-\sigma_{k^*} E)^{-1} E, (A-\sigma_{k^*} E)^{-1} b_l \big\} \Big\} \\
&\quad \bigcup \Big\{ \bigcup_{l=l^*+1}^{m} \mathcal{K}_{j^*-1}\big\{ (A-\sigma_{k^*} E)^{-1} E, (A-\sigma_{k^*} E)^{-1} b_l \big\} \Big\}.
\end{aligned} \tag{2.3}
$$

Note that this corresponds to the equality (2.2) and our description above of how V is built up, and also that a union where the lower index is larger than the upper index is defined to be empty.

For convenience, we collect the indices in the triplet $\{k, j, l\}$, in this order since Algorithms 1–2 form three nested loops with k as the index in the outermost loop and l as the index in the innermost loop. Using this notation, we refer to the step above as step $\{k^*, j^*, l^*\}$.

The integer s is a counter of the number of passes through the innermost loop. If no deflation has occurred we have $s = r$, but if vectors have been deflated we have $s > r$. This counter is introduced to facilitate the proof. For example, with one deflation at step $\{k, j, l\}$ we get $r = s - (J_b(k) - j + 1)$ in the end. It is clear that the equality we want to prove for $\operatorname{span}\{V\}$ is obtained for $\{k^*, j^*, l^*\} = \{K, J_b(K), m\}$, corresponding to $s = m\sum_{k=1}^{K} J_b(k)$.

For $s = 1$, we have $\gamma v_1 = (A - \sigma_1 E)^{-1} b_1$, where $\gamma = \| (A - \sigma_1 E)^{-1} b_1\|$, so (2.3) holds with $\{k^*, j^*, l^*\} = \{1, 1, 1\}$.

Assume that (2.3) holds for $s = s^* - 1$ with the matrix V_{r-1}, and for $1 < s < s^* - 1$ with corresponding matrices. At the next step, $\{k^*, j^*, l^*\}$, the candidate vector $\widetilde{v}_r$ at line 9 in Algorithm 2 is formed according to

$$\tilde{v}_r = (A - \sigma_{k^*} E)^{-1} b_{l^*} - \sum_{i=1}^{r-1} \alpha_i v_i \tag{2.4}$$

or

$$\tilde{v}_r = (A - \sigma_{k^*} E)^{-1} E v_q - \sum_{i=1}^{r-1} \alpha_i v_i\,, \tag{2.5}$$

where the α_i are orthogonalisation coefficients.

Equation (2.4) is used for $j^* = 1$. This is the easier case, where the induction hypothesis gives

$$
\begin{aligned}
\operatorname{span}\{V_{r-1}\} &= \Big\{ \bigcup_{k=1}^{k^*-1} \bigcup_{l=1}^{m} \mathcal{K}_{J_b(k)}\big\{ (A-\sigma_k E)^{-1} E, (A-\sigma_k E)^{-1} b_l \big\} \Big\} \\
&\quad \bigcup \Big\{ \bigcup_{l=1}^{l^*-1} \mathcal{K}_{1}\big\{ (A-\sigma_{k^*} E)^{-1} E, (A-\sigma_{k^*} E)^{-1} b_l \big\} \Big\},
\end{aligned}
$$

and it is clear that

$$\widetilde{v}_r \in \mathcal{K}_1\big\{ (A - \sigma_{k^*} E)^{-1} E, (A - \sigma_{k^*} E)^{-1} b_{l^*} \big\} \bigcup \operatorname{span}\{V_{r-1}\}\,.$$

If $\|\widetilde{v}_r\| > 0$, the vector $\widetilde{v}_r$ is normalised and added to V_{r-1} to form V_r, and (2.3) holds for $s = s^*$ with V_r. Otherwise, $(A - \sigma_{k^*} E)^{-1} b_{l^*} \in \operatorname{span}\{V_{r-1}\}$ and (2.3) holds for $s = s^*$ with V_{r-1}.

If $j^* > 1$, equation (2.5) is used, and the induction hypothesis gives

$$\begin{aligned}\operatorname{span}\{V_{r-1}\} &= \Big\{ \bigcup_{k=1}^{k^*-1} \bigcup_{l=1}^{m} \mathcal{K}_{J_b(k)}\{ (A-\sigma_k E)^{-1} E, (A-\sigma_k E)^{-1} b_l \} \Big\} \\ &\quad \bigcup \Big\{ \bigcup_{l=1}^{l^*-1} \mathcal{K}_{j^*}\{ (A-\sigma_{k^*} E)^{-1} E, (A-\sigma_{k^*} E)^{-1} b_l \} \Big\} \\ &\quad \bigcup \Big\{ \bigcup_{l=l^*}^{m} \mathcal{K}_{j^*-1}\{ (A-\sigma_{k^*} E)^{-1} E, (A-\sigma_{k^*} E)^{-1} b_l \} \Big\}.\end{aligned}$$

What we need to prove is that the direction

$$\{ (A-\sigma_{k^*} E)^{-1} E \}^{j^*-1} (A-\sigma_{k^*} E)^{-1} b_{l^*}$$

is either added to V_{r-1} at the present step s^* to form V_r or already contained in $\operatorname{span}\{V_{r-1}\}$.

We note from the choice of the index q, the induction hypothesis for the step at which V_q was formed, and the deflation procedure, that v_q can be written $v_q = v_q^{(1)} + v_q^{(2)}$, where

$$\begin{aligned} v_q^{(1)} &\in \bigcup_{k=1}^{k^*-1} \bigcup_{l=1}^{m} \mathcal{K}_{J_b(k)}\{ (A-\sigma_k E)^{-1} E, (A-\sigma_k E)^{-1} b_l \}, \\ v_q^{(2)} &\in \Big\{ \bigcup_{l=1}^{l^*} \mathcal{K}_{j^*-1}\{ (A-\sigma_{k^*} E)^{-1} E, (A-\sigma_{k^*} E)^{-1} b_l \} \Big\} \\ &\quad \bigcup \Big\{ \bigcup_{l=l^*+1}^{m} \mathcal{K}_{j^*-2}\{ (A-\sigma_{k^*} E)^{-1} E, (A-\sigma_{k^*} E)^{-1} b_l \} \Big\}, \end{aligned}$$

and $v_q^{(2)}$ has a nonzero component in the direction

$$\{ (A-\sigma_{k^*} E)^{-1} E \}^{j^*-2} (A-\sigma_{k^*} E)^{-1} b_{l^*} .$$

From this decomposition of v_q, it is obvious that

$$\begin{aligned} (A-\sigma_{k^*} E)^{-1} E v_q^{(2)} &\in \Big\{ \bigcup_{l=1}^{l^*} \mathcal{K}_{j^*}\{ (A-\sigma_{k^*} E)^{-1} E, (A-\sigma_{k^*} E)^{-1} b_l \} \Big\} \\ &\quad \bigcup \Big\{ \bigcup_{l=l^*+1}^{m} \mathcal{K}_{j^*-1}\{ (A-\sigma_{k^*} E)^{-1} E, (A-\sigma_{k^*} E)^{-1} b_l \} \Big\} \end{aligned}$$

since we operate with the same matrix, and using Lemma 1 we see that

$$\begin{aligned} (A-\sigma_{k^*} E)^{-1} E v_q^{(1)} &\in \Big\{ \bigcup_{k=1}^{k^*-1} \bigcup_{l=1}^{m} \mathcal{K}_{J_b(k)}\{ (A-\sigma_k E)^{-1} E, (A-\sigma_k E)^{-1} b_l \} \Big\} \\ &\quad \bigcup \Big\{ \bigcup_{l=1}^{m} \mathcal{K}_{1}\{ (A-\sigma_{k^*} E)^{-1} E, (A-\sigma_{k^*} E)^{-1} b_l \} \Big\}. \end{aligned}$$

Collecting the facts above, we see that equation (2.5) can be written

$$\widetilde{v}_r = \gamma \{ (A-\sigma_{k^*} E)^{-1} E \}^{j^*-1} (A-\sigma_{k^*} E)^{-1} b_{l^*} - \sum_{i=1}^{r-1} \alpha_i v_i .$$

Thus, if $\|\widetilde{v}_r\| > 0$, the vector $\widetilde{v}_r$ is normalised and added to V_{r-1} to form V_r, and (2.3) holds for $s = s^*$ with V_r. Otherwise,

$$\left(A - \sigma_{k^*}E\right)^{-1} E\Big\}^{j^*-1} \left(A - \sigma_{k^*}E\right)^{-1} b_{l^*} \in \operatorname{span}\left\{V_{r-1}\right\}$$

and (2.3) holds for $s = s^*$ with V_{r-1}. This concludes the proof of equation (2.3) for $s = s^*$, and thereby the proof of the equation for $\operatorname{span}\{V\}$.

The second part of the proof, corresponding to Z, is analogous to the presented part for V. □

We note that Lemma 1 is also used in [35], where rational Krylov algorithms for single-input single-output systems are described, and a variant that avoids the restarting is advocated for being more stable numerically. In this variant, instead of introducing the new starting vector $\left(A - \sigma_2 E\right)^{-1} b$ as the first vector corresponding to shift σ_2, the new vector $\left(A - \sigma_2 E\right)^{-1} EVt$ is created from a linear combination of the previous vectors. We have tested this variant as well and obtained results similar to the ones from the restarting algorithm, but choose the latter since we found it easier to combine with deflation in the case of multiple starting vectors.

We conclude this section with the necessary details on the last part of Algorithm 1. Given a large state space system, we choose σ, J_b and J_c and use Algorithm 1 to generate a reduced order system, and we know from Theorem 1 and the result presented in Section 1.4 that a certain number of moments are matched at the shift points. The expected order $\widetilde{r}$ of the reduced order system follows directly from these choices of input parameter values. Deflation of one or more vectors in the construction of the basis matrices means that we get a basis matrix with less than $\widetilde{r}$ columns for the union of Krylov subspaces. Thus, if the same number of vectors are deflated in the constructions of V and Z, we get the desired moment matching with a reduced order model that is smaller than expected. We are satisfied with this since we want a model that is as small as possible while matching these moments. If, on the other hand, a different number of vectors are deflated we need to expand the smaller of V and Z so that they get the same number of columns. We do this at the end of Algorithm 1. At this stage, we have the factorisation of the matrix corresponding to the last shift σ_K available, so we try to generate the missing number of columns with a new call of the Arnoldi function using this factorisation. This means that we match more moments at the last shift point. Due to deflations of vectors at this shift point it may happen that no, or too few, basis vectors can be generated in this way. In that case we choose not to do a new, expensive, factorisation with another shift, but rather add random vectors to the basis. Finally, we remark that in this last discussion we have implicitly assumed that exact deflation is used, since that is an assumption in Theorem 1. If a small but nonzero deflation tolerance is used, we use the same algorithm and obtain an approximate moment matching.

2.2 Choice of Shift Points

For the use of the dual rational Arnoldi algorithm, Algorithm 1, we have to choose how many shift points to use, where to place them, and how many moments to match at each shift point.

In order to obtain a good approximation over a wide range of frequencies with a small model, we want to use several shift points. The cost associated with this is that we make a sparse LU-factorisation of the matrix $A - \sigma_k E$ for each shift σ_k. Thus, we would like to have a few well-placed shift points.

We aim at constructing a reduced order model with a transfer function $\widehat{H}(s)$ that approximates the transfer function $H(s)$ of the full order system, especially at the imaginary axis, where $s = i\omega$, so that the frequency responses of the systems are similar. Grimme [19] explains that in general, for the approximation of $H(i\omega)$ by $\widehat{H}(i\omega)$, an imaginary shift point gives fast convergence close to the shift while a real shift point gives a slower convergence in a wider region.

Hopefully, the user who applies the algorithm to a specific problem has some knowledge of the dynamics of the problem, and can use this to choose shift points. At least, the frequency range $[\omega_{min}, \omega_{max}]$ of interest should be known or guesstimated. Then, we use imaginary shift points evenly distributed in this interval, either linearly or logarithmically depending on the range of the interval. For example, with three shift points and the interval $[1, 10^6]$ we place the shift points at $10i$, $10^3 i$, and $10^5 i$.

We would like to use an estimate of the error $H(s) - \widehat{H}(s)$ in an adaptive choice of shift points. Unfortunately, we have not been able to derive such an estimate that is sufficiently sharp and cheaply computed. Therefore, all we do here is to mention some of the work done in this area that gives a couple of ideas and might be extended. Some work on error estimates and adaptive methods is presented by Grimme [19], and a further study of adaptive shift point selection is made by Grimme and Gallivan [20]. Gallivan, Grimme and Van Dooren [17] place interpolation points on the positive real axis and give an algorithm where the choice among these in each step is based on the error in the corresponding moments; Chu and Lee [7] use the same idea in a rational Arnoldi algorithm. For a Lanczos method with only one shift point, Feldmann and Freund [13] make a choice of shift point based on the frequency range of interest, and Bai and Ye [5] derive an error estimate. Finally, the work by Li [27], and Li and White [28], on the connection between rational Krylov methods and algorithms for solving Lyapunov equations might lead to a new understanding on how to choose shift points.

2.3 Real Systems with Complex Shift Points

If the state space matrices of the original system have nonzero imaginary parts we expect that the basis matrices will as well. It is, however, quite common that the state space matrices are real and that it is desired to arrive at a real reduced order system. This is then automatically fulfilled if the shift points are real numbers, but not if at least one of them has a nonzero imaginary part. We often use imaginary shifts to obtain moment matching at certain frequencies, and then we can construct real bases as described for the rational Krylov method by Ruhe [32]. The important observation there is that if all quantities except the shift are real then

$$\begin{aligned}
\operatorname{Re}\left\{(A-\sigma E)^{-1}Ex\right\} &= \frac{1}{2}\left[\left((A-\sigma E)^{-1} + (A-\overline{\sigma}E)^{-1}\right)Ex\right], \\
\operatorname{Im}\left\{(A-\sigma E)^{-1}Ex\right\} &= \frac{1}{2i}\left[\left((A-\sigma E)^{-1} - (A-\overline{\sigma}E)^{-1}\right)Ex\right].
\end{aligned}$$

This means that if two complex conjugate shifts are used, it is sufficient to operate with one of them, divide the resulting vector into its real and imaginary parts, and add these two real vectors to the basis. Note that by using $\sigma = i\omega$, $\omega \in \mathbb{R}$, we can make sure that the reduced order system has the same steady state response to a sinusoidal input of frequency ω as the full system. The eigenvalues of the reduced order system tend to those of the full order system, and the use of a shift enhances convergence towards eigenvalues close to the shift. Since the real system we consider for the moment has complex conjugate eigenvalues, it makes some sense to also use the shift $\sigma = -i\omega$, and this is what is done implicitly in our construction of real bases.

We use this method for constructing real bases for real systems in our implementation of Algorithms 1–2. We usually choose all the shifts to be imaginary, and then condition (2.1) can still be used for checking that the basis matrices are expected to get the same number of columns. Since we want to have the possibility of using a combination of real shifts and shifts with nonzero imaginary part, we replace (2.1) by

$$m \sum_{k=1}^{K} (1 + \chi_k) J_b(k) = p \sum_{k=1}^{K} (1 + \chi_k) J_c(k),$$

where $\chi_k = 1$ if $\mathrm{Im}\{\sigma_k\} \neq 0$ and $\chi_k = 0$ otherwise. This is the natural condition obtained from directly comparing the expected sizes of the two basis matrices, since we implicitly also use the complex conjugate of each shift with a nonzero imaginary part.

2.4 Factorisation or Iterative Solver

We need to solve linear systems with the matrices $A - \sigma_k E$ to generate basis matrices for the desired union of Krylov subspaces (1.5). If these linear systems are solved to full accuracy, the moments of the reduced order system will match those of the full order system, so that is what we do. We first describe how this is done using a sparse factorisation, and then briefly discuss the possibility of using an iterative solver instead.

We make an LU-factorisation of the matrix $A - \sigma_k E$, for each shift σ_k, in Algorithm 1. The number of nonzero entries in the LU-factors of a matrix can depend strongly on how the rows and columns are ordered. Unfortunately, the problem of finding how to reorder the rows and columns of a matrix to obtain optimally sparse LU-factors is NP-complete—for those not familiar with this class of problems, in this context it is enough to know that it means that there is no known algorithm that solves the problem in a time that is polynomial in the matrix size—so we have to use some heuristic method for reordering the matrix in a way that is at least reasonably good. Several such heuristics are available in MATLAB, including, for example, minimum degree reorderings from the functions *symmmd* and *colmmd* and reverse Cuthill–McKee reorderings from the function *symrcm*; see [18] for descriptions of these methods. These functions return a permutation vector, that is, a vector of the integers $1, 2, \ldots, n$, with which the matrix is reordered. Some of the functions are mainly intended for reordering of both the rows and the columns of symmetric matrices, while others are made for reordering the columns of nonsymmetric matrices. All these functions determine a permutation based on the sparsity structure of the matrix, without considering the numerical values of the nonzero elements. Therefore, we choose one of the functions and determine a permutation for $|A| + |E|$, and then we use this single permutation for all the matrices $A - \sigma_k E$. We believe that this is the best we can do, but note that pivoting for stability may be done in the LU-factorisation, depending on the numerical values of the nonzero entries.

We use the symmetric minimum degree reordering function *symmmd* for symmetric matrices and nonsymmetric matrices with symmetric sparsity patterns, while the column approximate minimum degree reordering from *colamd* applied to both the columns and the rows is our choice for general nonsymmetric matrices. In Chapter 4, we show the effects of the reorderings we have used on the problems presented there.

For each reordered matrix, we use the MATLAB function *luinc* to obtain sparse LU-factors. This function can be used for producing various incomplete factorisations, but with the drop tolerance set to zero, it gives us a complete factorisation.

If the matrix size n is very large, the LU-factorisation may take a long time and require a lot of memory for storing the factors. In such cases, use of an iterative method for solving the linear systems should be considered. We have implemented the optional use of an iterative solver instead of the LU-factors for each operation with an inverted matrix. There are several iterative solvers and preconditioners available in FEMLAB; see the entry *femiter* in the reference manual [9] for details. The best choice of solver and preconditioner is highly problem dependent, and something we have not looked at in detail. We advocate that the systems are solved to full accuracy so that the desired moment matching is obtained. Grimme [19] discusses solving the systems to different levels of accuracy, and if they are not solved to full accuracy the operator should be of the form $(A - \sigma E)^{-1}(A - \zeta E)$ instead of $(A - \sigma E)^{-1}E$, where the choice of the scalar ζ is nontrivial. These approximate solves are also discussed by Grimme et al. [21]. This is also related to the work by Lehoucq and Meerbergen [26] on approximate solves in the rational Krylov method for computing eigenvalues, where the Cayley transformation $(A - \sigma E)^{-1}(A - \zeta E)$ is shown to give a better performance of the algorithm than the shift-invert transformation $(A - \sigma E)^{-1}E$ does.

Chapter 3

Stabilisation

The reduced order system (1.2) we obtain from a stable full order system (1.1) using bases from the dual rational Arnoldi algorithm may be unstable, that is, the matrix pencil $\widehat{A} - \lambda\widehat{E}$ may have eigenvalues with positive real part. When we want to do a time-stepping of the reduced order system, we need to modify it to get a stable system. We have to do this in such a way that the stabilised system is still a good approximation to the original system.

3.1 Additive Decomposition

We choose to do an additive decomposition of the transfer matrix $\widehat{H}(s)$ into a stable part $\widehat{H}_s(s)$ and an unstable part $\widehat{H}_u(s)$, and use only the stable part for simulation, thus obtaining a smaller system with eigenvalues equal to the stable eigenvalues of the unstable system. Here we describe this for the most common case of real state space matrices; the decomposition for complex state space matrices is basically the same, but a bit simpler since it is done with a complex Schur decomposition without the 2×2 diagonal blocks.

Starting from the unstable matrix pencil $\widehat{A} - \lambda\widehat{E}$ we first make a real Schur decomposition using the MATLAB function *qz* with option 'real', so that

$$Q^T(\widehat{A} - \lambda\widehat{E})Z = S - \lambda T,$$

where Q and Z are orthogonal, T is upper triangular, and S is upper quasi-triangular with a 2×2 diagonal block for each pair of complex conjugate eigenvalues. We then reorder the matrices so that the unstable part is put in the lower right corner:

$$X^T(S - \lambda T)Y = \widetilde{S} - \lambda\widetilde{T} = \begin{pmatrix} \widetilde{S}_{11} & \widetilde{S}_{12} \\ 0 & \widetilde{S}_{22} \end{pmatrix} - \lambda \begin{pmatrix} \widetilde{T}_{11} & \widetilde{T}_{12} \\ 0 & \widetilde{T}_{22} \end{pmatrix},$$

where X and Y are orthogonal, the pencil $\widetilde{S}_{11} - \lambda\widetilde{T}_{11}$ is stable and the pencil $\widetilde{S}_{22} - \lambda\widetilde{T}_{22}$ is anti-stable. The method we use for shifting the positions of two adjacent diagonal blocks is described by Kågström and Poromaa [23] and based on the solution of a small generalised Sylvester equation. We note that Brandts [6] discusses, and gives MATLAB code for, sorting a real Schur form of a matrix, although we have not used his implementation. Finally, we decouple the stable and unstable parts by the transformation

$$
\begin{aligned}
P^{-1}(\widetilde{S} - \lambda \widetilde{T})N &= \begin{pmatrix} I & -L \\ 0 & I \end{pmatrix} \left\{ \begin{pmatrix} \widetilde{S}_{11} & \widetilde{S}_{12} \\ 0 & \widetilde{S}_{22} \end{pmatrix} - \lambda \begin{pmatrix} \widetilde{T}_{11} & \widetilde{T}_{12} \\ 0 & \widetilde{T}_{22} \end{pmatrix} \right\} \begin{pmatrix} I & R \\ 0 & I \end{pmatrix} \\
&= \begin{pmatrix} \widetilde{S}_{11} & 0 \\ 0 & \widetilde{S}_{22} \end{pmatrix} - \lambda \begin{pmatrix} \widetilde{T}_{11} & 0 \\ 0 & \widetilde{T}_{22} \end{pmatrix},
\end{aligned}
$$

where L and R are the solutions of the generalised Sylvester equation

$$
\begin{aligned}
\widetilde{S}_{11}R - L\widetilde{S}_{22} &= -\widetilde{S}_{12}\,, \\
\widetilde{T}_{11}R - L\widetilde{T}_{22} &= -\widetilde{T}_{12}\,,
\end{aligned}
$$

which has a unique solution since the pencils $\widetilde{S}_{11} - \lambda\widetilde{T}_{11}$ and $\widetilde{S}_{22} - \lambda\widetilde{T}_{22}$ have disjoint spectra. This is described by Kågström and Westin [24], and we use their algorithm for solving the equation. With the matrix

$$
P_{\text{stab}} = \begin{pmatrix} I \\ 0 \end{pmatrix}
$$

for extracting the upper left blocks of the matrices, we can combine the steps above and arrive at

$$
\widetilde{S}_{11} - \lambda\widetilde{T}_{11} = P_{\text{stab}}^T P^{-1} X^T Q^T (\widehat{A} - \lambda\widehat{E}) ZYNP_{\text{stab}} \equiv Z_{\text{stab}}^T (A - \lambda E) V_{\text{stab}}\,.
$$

This means that the stable reduced order system can be written on the same form as the unstable reduced order system (1.2), but it has lower order $\widetilde{r} < r$. The matrices V and Z are replaced by

$$
\begin{aligned}
V_{\text{stab}} &= VZYP_{\text{stab}}\,, \\
Z_{\text{stab}} &= ZQXP^{-T}P_{\text{stab}}\,,
\end{aligned}
\tag{3.1}
$$

where we note that $V_{\text{stab}}^T V_{\text{stab}} = I$ so V_{stab} is orthogonal, while $Z_{\text{stab}}^T Z_{\text{stab}} = I + LL^T$ so Z_{stab} is not orthogonal.

Unfortunately, we cannot expect the matrices V_{stab} and Z_{stab} to satisfy the inclusions (1.5) of the Krylov subspaces, so the stabilised system may not have the moment matching properties of the unstable system. Since we make the decomposition $\widehat{H}(s) = \widehat{H}_s(s) + \widehat{H}_u(s)$, the change in the transfer function is given by $\widehat{H}_u(s)$. We can cheaply evaluate this error at a number of frequencies and give a warning if it is large. In such cases, the user may decide to start over and generate another reduced order model of a different size or using different shift points.

Finally, we remark that it is explained in [23] that sometimes a swap of two diagonal blocks should not be made, because it would introduce an unacceptable error in the resulting pencils. They also give stability tests for checking if a swap should be accepted or not. We need to make all the swaps in order to arrive at the desired stable pencil, but we use one of the stability tests for warning the user when a swap may have lead to an inaccurate result.

3.2 Other Stabilisation Ideas

The additive decomposition, described in the previous section, corresponds to discarding the unstable modes of the reduced order system to obtain a stable system of lower order.

There is some work that indicates that this can be a reasonable thing to do, and other work on alternative stabilisation methods, that we point out here.

Feldmann and Freund [13] use a Lanczos method with one shift point. In the choice of shift point, the aim is that the reduced order model will be a good approximation in a given frequency range. This model is given in terms of a pole/residue representation of the transfer function. The model is unstable if there are positive poles, and any such poles are deleted when a stable model is needed. They state that this is safe to do as long as the original model is stable and the reduced order model has converged in the frequency range of interest, since then the unstable poles must either have small residues or be far from the frequency range considered.

Grimme, Sorensen and Van Dooren [22] derive an implicitly restarted Lanczos method for standard state space systems. They use the unstable poles as shifts for the restarts, and show that the resulting reduced order model matches a number of modified moments where the starting vectors are replaced.

Jaimoukha and Kasenally [25] give an implicitly restarted dual Arnoldi method for standard state space systems. They show that while a model obtained without restarts matches a maximal number of moments, their use of implicit restarts gives a model that matches some of these moments. They also explain how to do an implicit restart in which the unstable part of the reduced order model is removed.

Bai and Freund [4] give an algorithm for a Lanczos method with one shift point, and with postprocessing to obtain a stable reduced order model. In the postprocessing, the reduced order model is modified to have a number of prescribed poles and zeros. The result is a model of the same size that matches fewer moments but is stable and possibly passive. As an example, they stabilise a model by prescribing poles at $-\mathrm{Re}\{\lambda\} + i\mathrm{Im}\{\lambda\}$ for each unstable pole λ; in other words, unstable poles are reflected in the imaginary axis. When the unstable poles are moved in this way, the stable poles may also be affected, so the process has to be iterative. They point out that it is not clear how to prescribe poles and zeros for generating a stable reduced order model that matches a maximal number of moments.

Chapter 4

Numerical Results

Here we present the application of our code to two test problems in FEMLAB. We have used FEMLAB 2.3 together with MATLAB 6.1 under Linux on a PC with 2.10 GB memory and two Pentium III 993 MHz processors with 256 kB cache.

4.1 Convection Problem

We consider a convection problem in two space dimensions, on the domain shown in Figure 4.1. The equation to solve is a partial differential equation with an inflow boundary

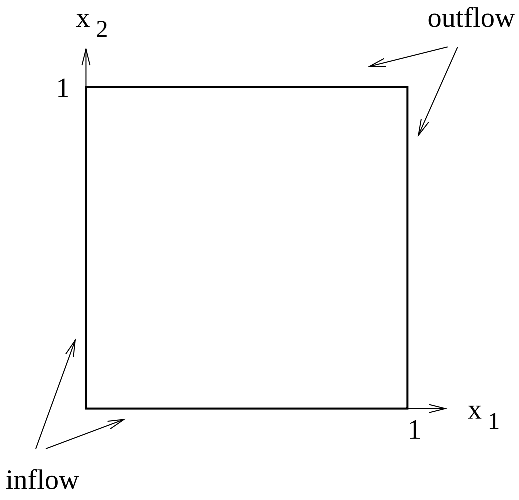


Figure 4.1: Space domain of the convection problem.

condition and an initial condition:

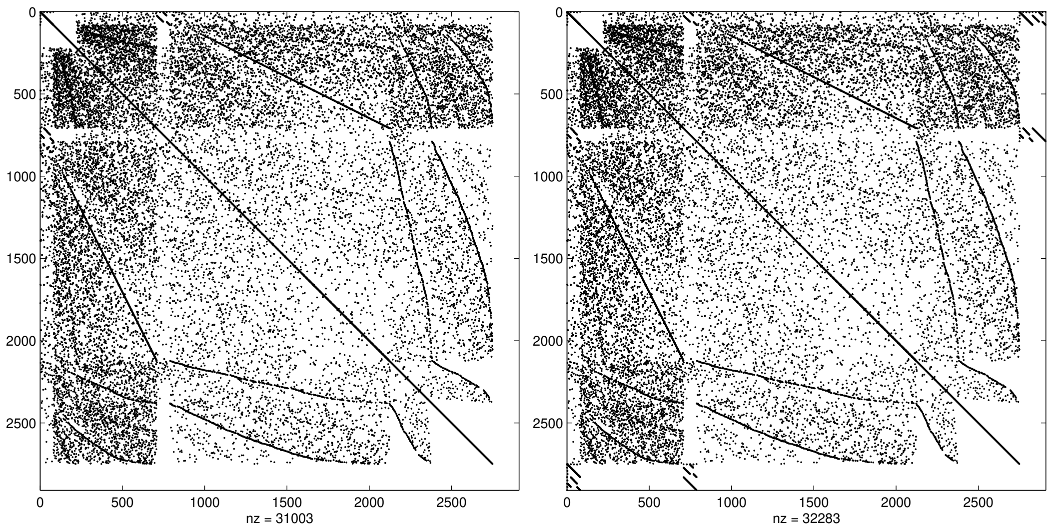
$$\begin{cases} \dot{u} + \beta \cdot \nabla u = 0 & \text{in } (0,1)\times(0,1) \text{ for } t>0, \\ u = (1-x_1-x_2)u_{in} & \text{at } \{x_1=0, 0\le x_2 \le 1\} \bigcup \{0 \le x_1 \le 1, x_2 = 0\}, \\ u = 0 & \text{in } (0,1)\times(0,1) \text{ for } t=0. \end{cases}$$

Here, x_1 and x_2 are the two space variables and t is the time variable. The vector β is known as the convection velocity, and we let $\beta = (1,1)$, while the scalar u_{in} in the boundary condition is an input variable.

For stability reasons, we add a small mesh-dependent amount of streamline diffusion explicitly in the weak formulation of the equation.

We impose the boundary condition in a weak form, and in the same way we set $u = 0$ on the outflow boundary. For the finite element discretisation, we use second order Lagrange elements; see the reference manual [9].

We create a state space system of the form (1.1) starting from the stationary solution $u = 0$ in the entire domain for $u_{in} = 0$. As input variable we choose u_{in} as mentioned above, and as output variables we choose the value of u at a mesh point close to $(x_1, x_2) = (0.9, 0.7)$ and the integral of u over the domain. This full order system has 2909 degrees of freedom. In Figure 4.2, we see the sparsity structures of the state space matrices. We point out that the



(a) E.

(b) A.

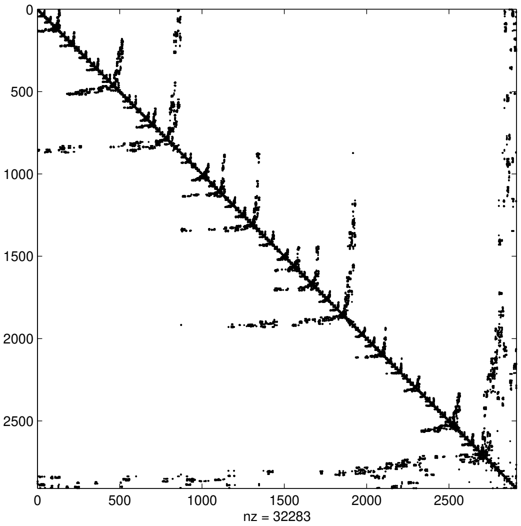
Figure 4.2: Sparsity structures of the state space matrices in the convection problem.

matrix A is not normal, although its sparsity structure is symmetric.

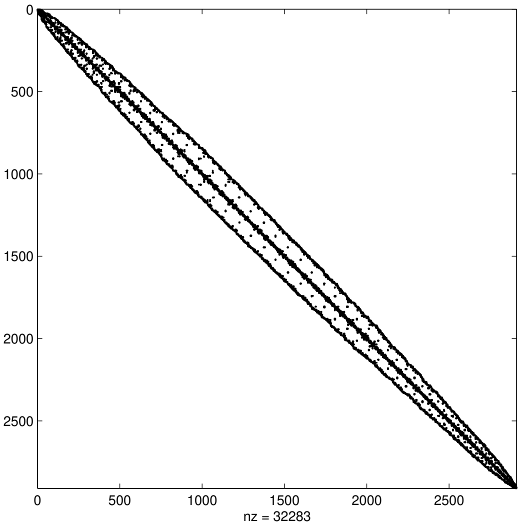
We generate a reduced order model with the dual rational Arnoldi algorithm. To get a good approximation of the transfer function $H(s)$ for $s = i\omega$, $\omega \in [10^{-2}, 10^{4}]$, at a reasonable cost, we use three imaginary shift points logarithmically spaced in this interval, and for each shift we generate four basis vectors from each output vector and eight basis vectors from the input vector. Thus, we obtain a reduced order model with 48 degrees of freedom.

With rows and columns reordered using the permutations from *symmmd* and *symrcm*, the matrix $A - \sigma_1 E$ has the sparsity structure seen in Figure 4.3. The LU-factors of these reordered matrices have a total of 134549 and 346161 nonzero elements, respectively, so for this problem it is clear that the minimum degree algorithm works better than the bandwidth reducing reverse Cuthill–McKee algorithm. To indicate the importance of reordering the matrix, we remark that the LU-factors of the matrix in its original form have a total of 6278306 nonzero elements. The matrix itself has 32283 nonzero elements and 8429998 zero elements.

For comparison with the dual rational Arnoldi reduced order model, we use the mode reduction described at the end of Section 1.3 to generate an alternative model of the same size, from 47 eigenvectors and the static solution vector. We note that this involves the factorisa-



(a) $A - \sigma_1 E$, reordered with *symmmd*.



(b) $A - \sigma_1 E$, reordered with *symrcm*.

Figure 4.3: Sparsity structure of the first matrix factorised in the dual rational Arnoldi algorithm applied to the convection problem, reordered using symmetric minimum degree and reverse Cuthill–McKee algorithms.

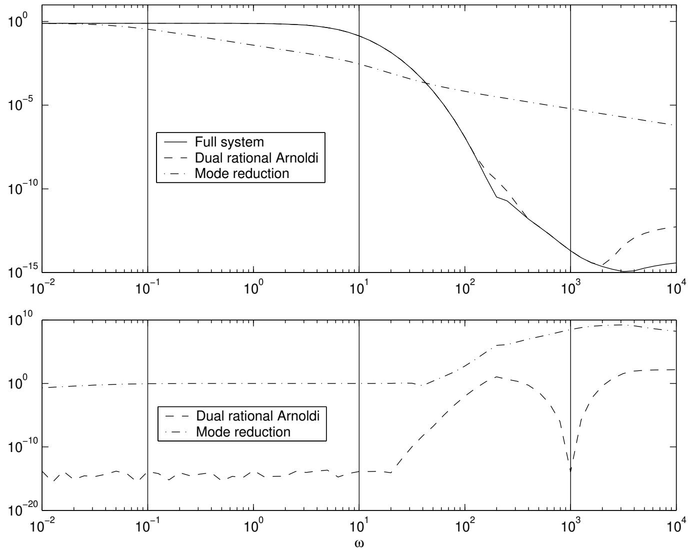
tion of the matrix $A - 0E$ in a spectral transformation Arnoldi algorithm. This is an iterative algorithm with a convergence criterion, since it is used for computing eigenvectors. On the other hand, in our use of the dual rational Arnoldi algorithm described above, we factorise three matrices $A - \sigma_k E$ and the Arnoldi algorithm is a means of computing orthogonal basis vectors.

The transfer functions of the systems are shown in Figure 4.4. We see that the transfer function is rather well approximated with the dual rational Arnoldi algorithm, except at the highest frequencies, but very poorly approximated with the mode reduction. We remark that the shift points are chosen in an automatic way and the same number of moments are matched at each shift; a better model could have been obtained if some prior knowledge had made us match more moments in the high frequency region.

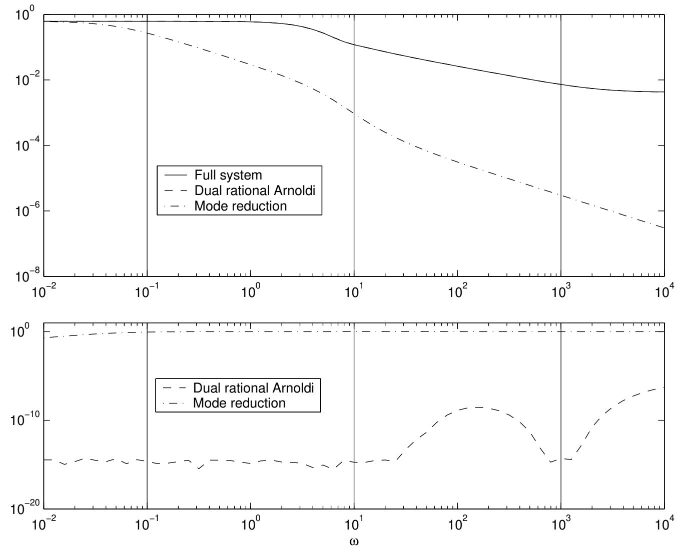
The transfer functions of different models can also be compared in Nyquist diagrams, where $\mathrm{Im}\{H(i\omega)\}$ is plotted versus $\mathrm{Re}\{H(i\omega)\}$. In Figure 4.5, we see these diagrams for ω in the same interval as in Figure 4.4.

It is interesting to look at the eigenvalues of the reduced order pencils $\widehat{A} - \lambda\widehat{E}$; see Figure 4.6. The one eigenvalue that is not at all visible in this figure belongs to the dual rational Arnoldi reduced pencil and is located at $-9.84 * 10^9$. We note that some eigenvalues close to the origin coincide, and that the dual rational Arnoldi reduced pencil is unstable since it has one eigenvalue in the right half plane. The eigenvalues of this pencil cover a larger part of the left half plane. Thus, there are poles with relatively high damping, that is, a larger negative real part, that are important to the input-output behaviour of the system. The presence of an eigenvalue very close to the origin in the pencil from mode reduction is caused by the inclusion of the static mode.

Finally, we compare the results from time-stepping the original and dual rational Arnoldi

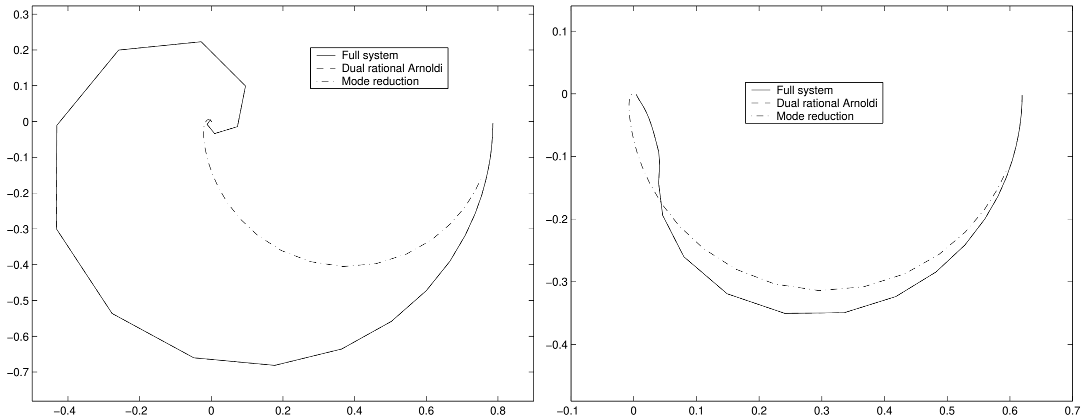


(a) $H_{11}(i\omega)$.



(b) $H_{21}(i\omega)$.

Figure 4.4: Amplitudes of the entries of the transfer functions for the full order model and its approximations in the convection problem. The upper plots show the amplitudes $|H_{jk}(i\omega)|$; the lower plots show the relative errors $|H_{jk}(i\omega) - \widehat{H}_{jk}(i\omega)|/|H_{jk}(i\omega)|$. The vertical lines indicate the shift points, that is, they are placed for ω such that $\sigma = i\omega$ was used as a shift point in the dual rational Arnoldi algorithm.



(a) $H_{11}(i\omega)$.

(b) $H_{21}(i\omega)$.

Figure 4.5: Nyquist diagrams of the transfer functions for the full order model and its approximations in the convection problem.

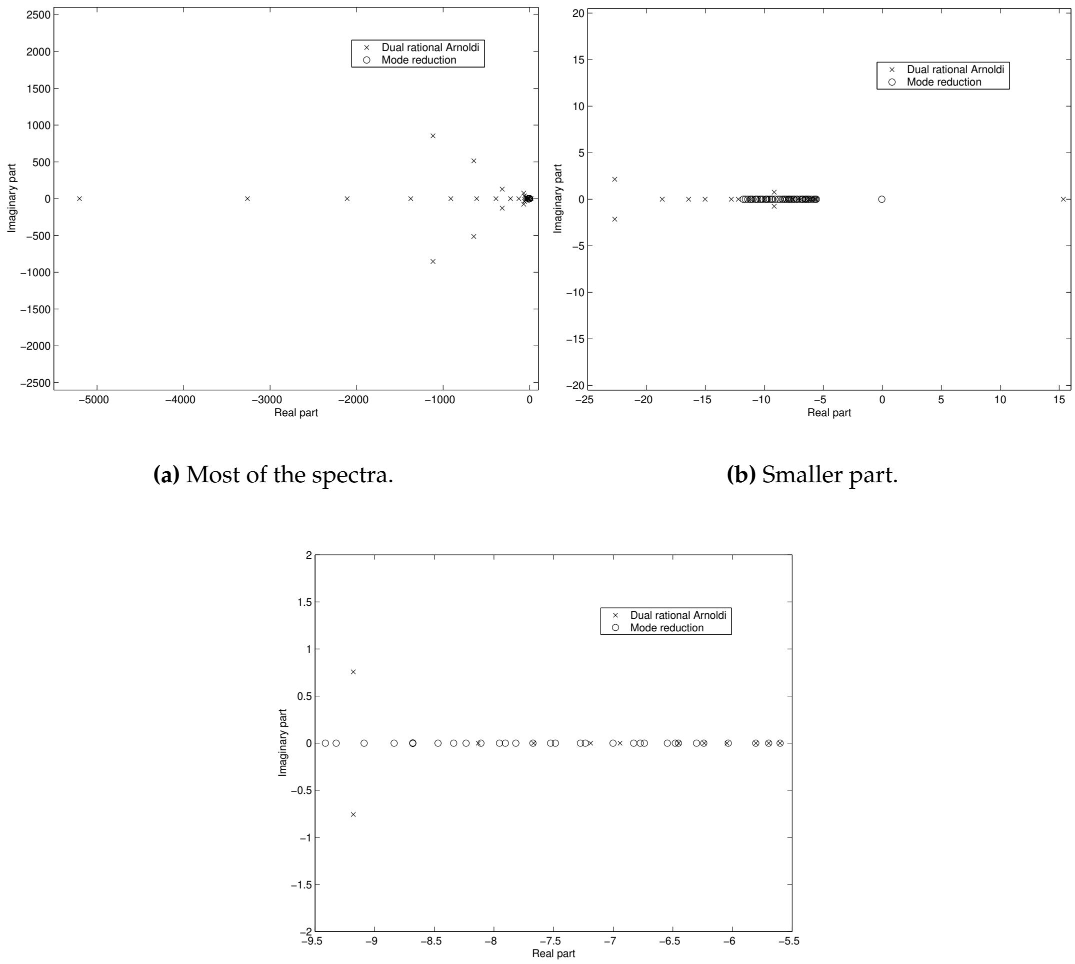
reduced order systems. For this comparison we use the original FEMLAB model, and not the full order state space system, since that is what is naturally used in FEMLAB.

As the reduced order pencil is unstable, we must first stabilise it. There was one unstable eigenvalue and so we end up with a stable system with 47 degrees of freedom. We use the function *ode15s* in MATLAB for time-stepping the reduced order system. This is a function for integrating stiff differential equations and differential-algebraic equations over a time interval, using a method of variable order based on the numerical differentiation formulas; see [34] for details. For time-stepping the original system, we use the standard solver *femtime* in FEMLAB, which by default utilises *ode15s*; see [9]. The output signals $\widehat{y}(t)$ and $y(t)$ of the reduced order system and the original system, respectively, can be compared directly. The reduced order state $\widehat{x}(t)$ is multiplied by the basis matrix V_{stab} to give the approximate state $V_{\text{stab}}\widehat{x}(t)$, and further multiplied by the FEMLAB constraint handling matrix Null to give the approximate solution $\text{Null} \cdot V_{\text{stab}}\widehat{x}(t)$. With the input $u_{in} = 1 + 0.5\sin(5t)$ we get the results shown in Figures 4.7–4.9. It is clear from these pictures that the reduced order model is a very good approximation in this case. The size of the error in the second output at $t = 0$ in Figure 4.7 is 0.0043. The reason for this error is that the original system starts from $u = u_{in} = 1$ at the inflow boundary while the reduced order system starts from $u = 0$ there.

4.2 Vibrating Machines

This problem is taken from the model library of the FEMLAB Structural Mechanics Module [8]. It is a model of a steel floor on which two dynamic machines and a sensitive instrument are placed; see Figure 4.10. The first machine stands on two circular foundations, and the second machine stands on one circular foundation. We are interested in determining the influence of the dynamic machines on the floor, especially at the location of the instrument.

The floor is modelled as a simply supported plate, with increased density at the machine foundations to account for the weight of the machines. The finite element discretisation is



(a) Most of the spectra. **(b)** Smaller part.

(c) Close to the origin.

Figure 4.6: Eigenvalues of the reduced order pencils in the convection problem.

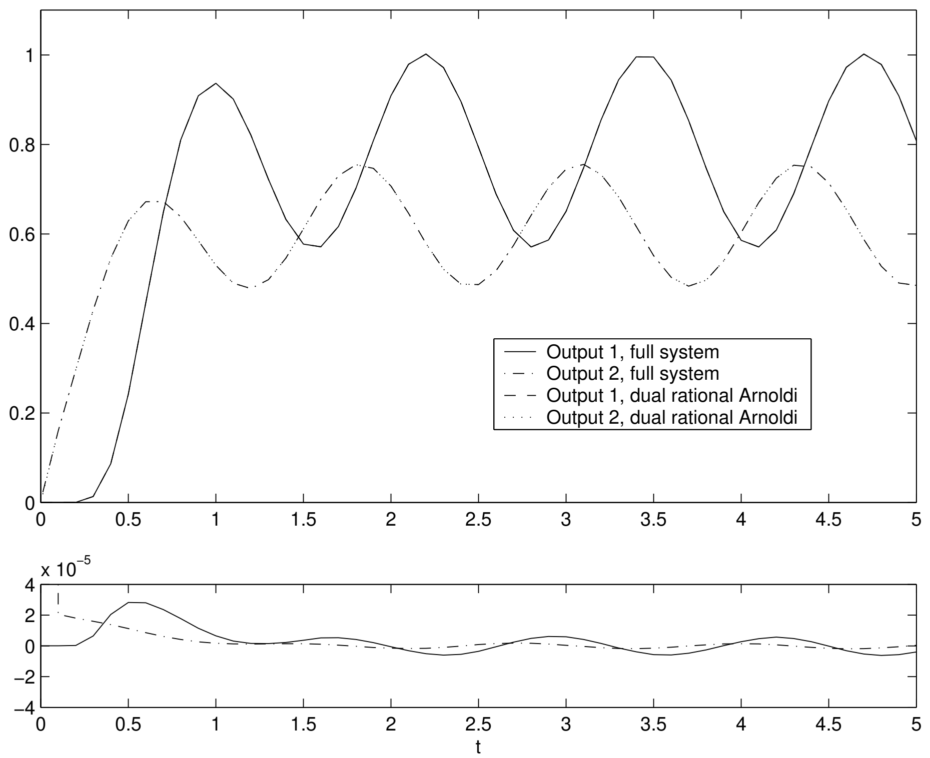
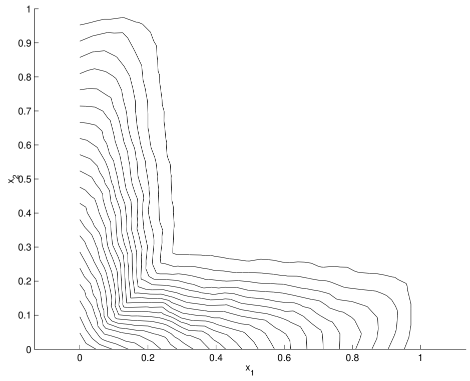
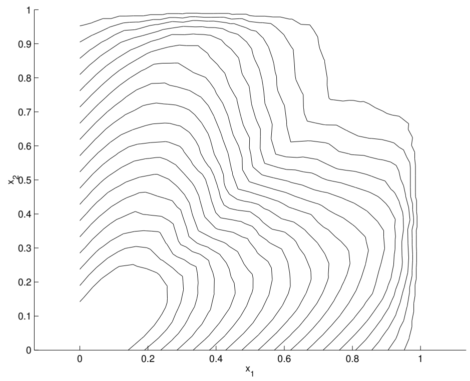


Figure 4.7: Output values, $y(t)$ for the original model and $\widehat{y}(t)$ for the dual rational Arnoldi reduced order system, in the convection problem. The lower graph shows the differences $y_1(t) - \widehat{y}_1(t)$ (solid) and $y_2(t) - \widehat{y}_2(t)$ (dashdot).



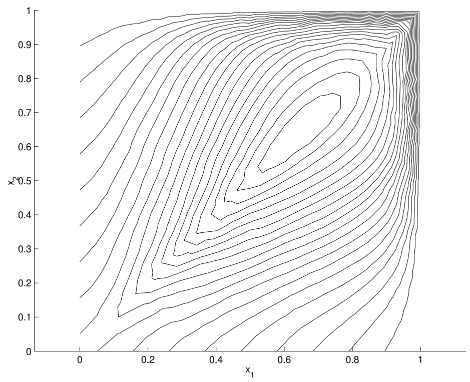
(a) Original system, $t = 0.1$.

(b) Dual rational Arnoldi, $t = 0.1$.



(c) Original system, $t = 0.4$.

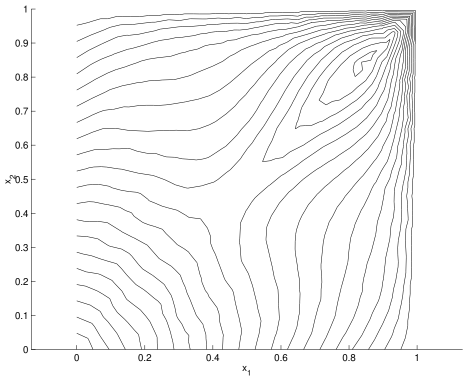
(d) Dual rational Arnoldi, $t = 0.4$.



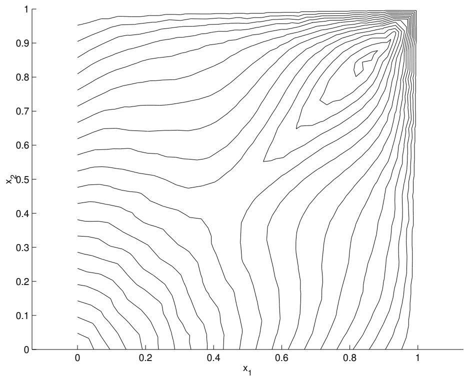
(e) Original system, $t = 1$.

(f) Dual rational Arnoldi, $t = 1$.

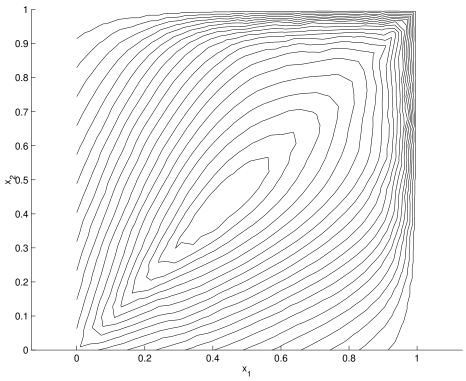
Figure 4.8: Contour plots of the solutions from the time-steppings of the convection problem, 20 levels from 0 to 1.5.



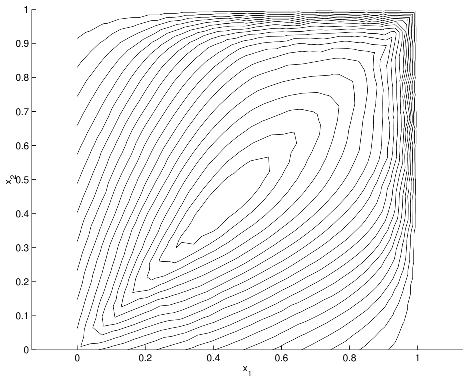
(a) Original system, $t = 1.4$.



(b) Dual rational Arnoldi, $t = 1.4$.



(c) Original system, $t = 2$.



(d) Dual rational Arnoldi, $t = 2$.

Figure 4.9: Contour plots of the solutions from the time-steppings of the convection problem, 20 levels from 0 to 1.5.

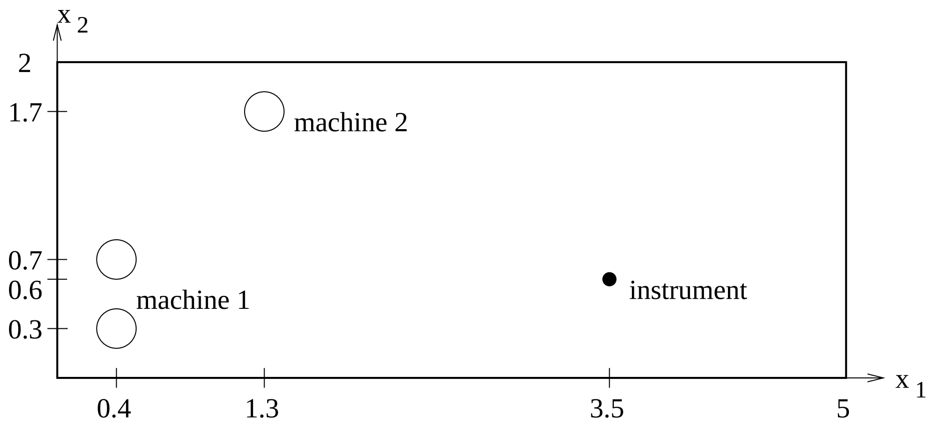
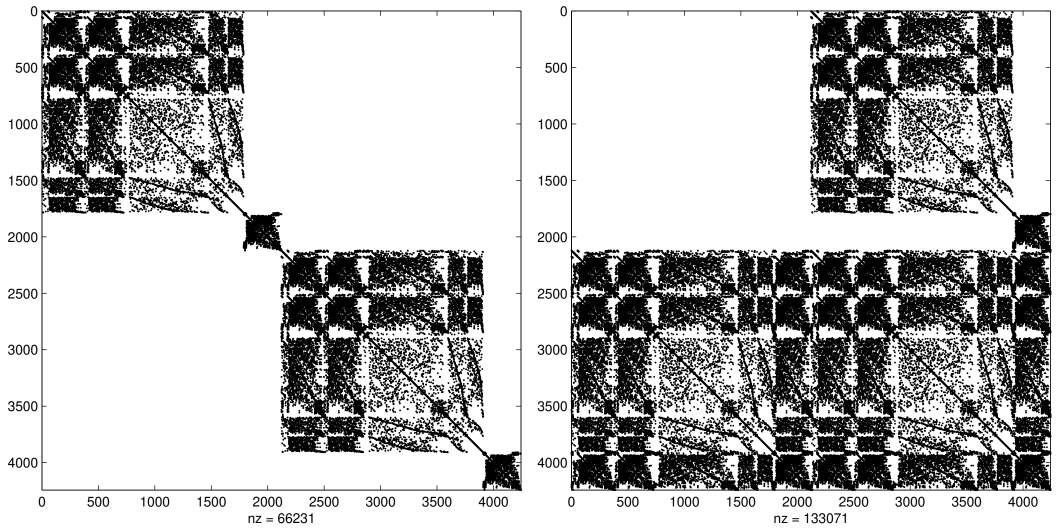


Figure 4.10: Setup of the vibrating machines problem. The radius of each circular foundation is 0.1 units.

made with the mesh seen in Figure 1.1 and a discrete Reissner–Mindlin plate element with twelve degrees of freedom; see [8] for details on this element type. We note from Figure 1.1 that the mesh is finer close to the locations of the machines and the instrument and coarser away from these areas. Furthermore, wee see that the mesh is overall rather coarse. This means that it does not seem reasonable to use a much coarser mesh to obtain a system of lower order.

For the state space system, we choose the loads from the machines as input variables and the displacement in the vertical (x_3) direction at the instrument location as output variable. This full order system has 4242 degrees of freedom. In Figure 4.11, we see the sparsity structures of the state space matrices. The original partial differential equation contains second



(a) E.

(b) A.

Figure 4.11: Sparsity structures of the state space matrices in the vibrating machines problem.

order time derivatives and is transformed into a first order system through the introduction of additional variables; the effect of this is seen in the structure of the state space matrices. As an illustrative example, we consider the equation $M\ddot{x} + D\dot{x} + Kx = 0$, which by the introduction of the new variable $v = \dot{x}$ can be rewritten as

$$\begin{pmatrix} I & 0 \\ 0 & M \end{pmatrix} \begin{pmatrix} \dot{x} \\ \dot{v} \end{pmatrix} + \begin{pmatrix} 0 & -I \\ K & D \end{pmatrix} \begin{pmatrix} x \\ v \end{pmatrix} = 0.$$

In the model library in [8], this vibrating machines model is studied for loads of frequencies $\omega \in [2\pi \cdot 50, 2\pi \cdot 125]$. We consider frequencies in the somewhat larger interval $[100, 1000]$, but since this frequency range is still only one order of magnitude we use no more than two shift points. With our standard choice of imaginary shift points logarithmically spaced in the interval, for each shift we generate two basis vectors from each input vector and four basis vectors from the output vector, and obtain a reduced order model with 16 degrees of freedom.

The sparse matrix $A - \sigma_1 E$ is not symmetric, so we use the column approximate minimum degree permutation from *colamd* to reorder the rows and columns prior to the LU-factorisation. The matrix has 166191 nonzero elements, and the sparsity structure shown in Figure 4.12 after reordering, while the LU-factors have a total of 1257715 nonzero elements.

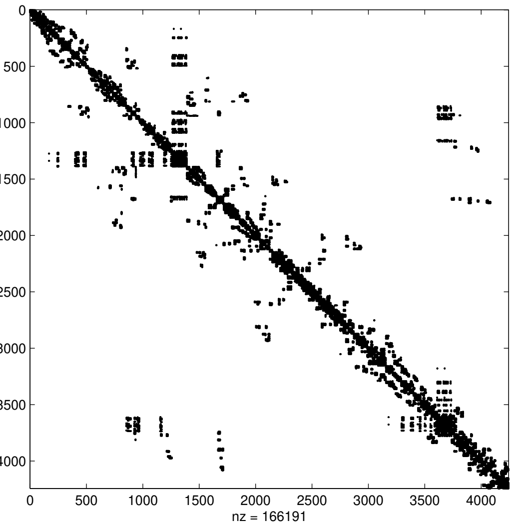


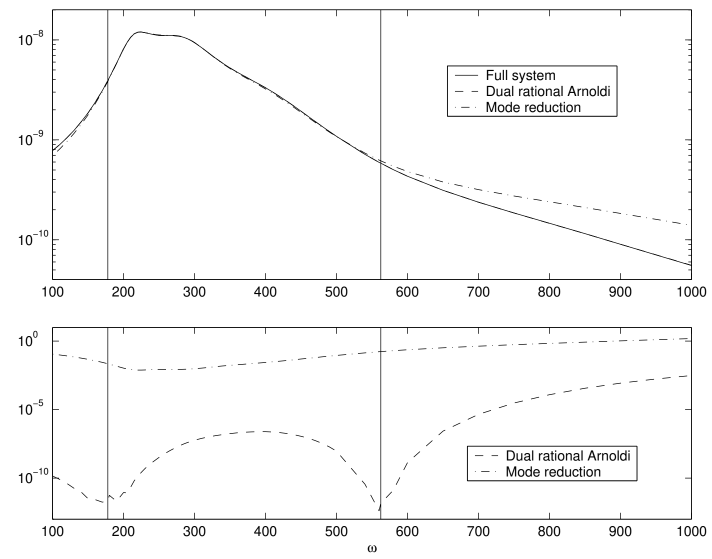
Figure 4.12: Sparsity structure of the first matrix factorised in the dual rational Arnoldi algorithm applied to the vibrating machines problem, reordered using a column approximate minimum degree permutation.

With the mode reduction, we generate an alternative model from 14 eigenvectors and the two static solution vectors.

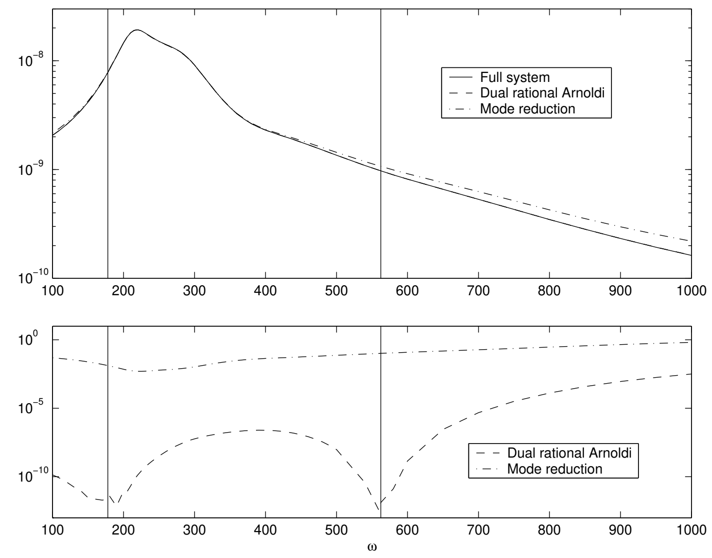
The transfer functions of the systems are shown in Figure 4.13. We see that the transfer function is very well approximated with the dual rational Arnoldi algorithm, and reasonably well approximated at low frequencies with the mode reduction. The difference between the approximations is less visible in a Nyquist diagram; see Figure 4.14.

The eigenvalues of the reduced order pencils $\widehat{A} - \lambda\widehat{E}$ are shown in Figure 4.15. The two eigenvalues close to zero in the pencil from mode reduction correspond to the static modes.

Both reduced order models are stable and seem to be useful approximations. We simulate the full and reduced order models with the loads $1000\sin(550\pi t/3)/3$ on both inputs; this combination of loads appear in the presentation in the model library. The resulting output is shown in Figure 4.16, where we see that both approximations are quite good, although the result from the dual rational Arnoldi model is somewhat better.

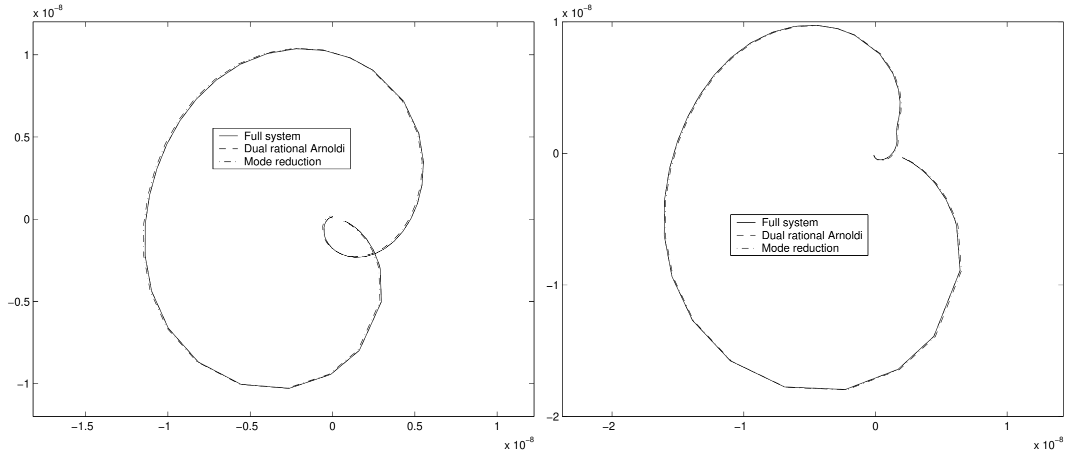


(a) $H_{11}(i\omega)$.



(b) $H_{12}(i\omega)$.

Figure 4.13: Amplitudes of the entries of the transfer functions for the full order model and its approximations in the vibrating machines problem. The upper plots show the amplitudes $|H_{jk}(i\omega)|$; the lower plots show the relative errors $|H_{jk}(i\omega) - \widehat{H}_{jk}(i\omega)|/|H_{jk}(i\omega)|$. The vertical lines indicate the shift points, that is, they are placed for ω such that $\sigma = i\omega$ was used as a shift point in the dual rational Arnoldi algorithm.



(a) $H_{11}(i\omega)$.

(b) $H_{12}(i\omega)$.

Figure 4.14: Nyquist diagrams of the transfer functions for the full order model and its approximations in the vibrating machines problem

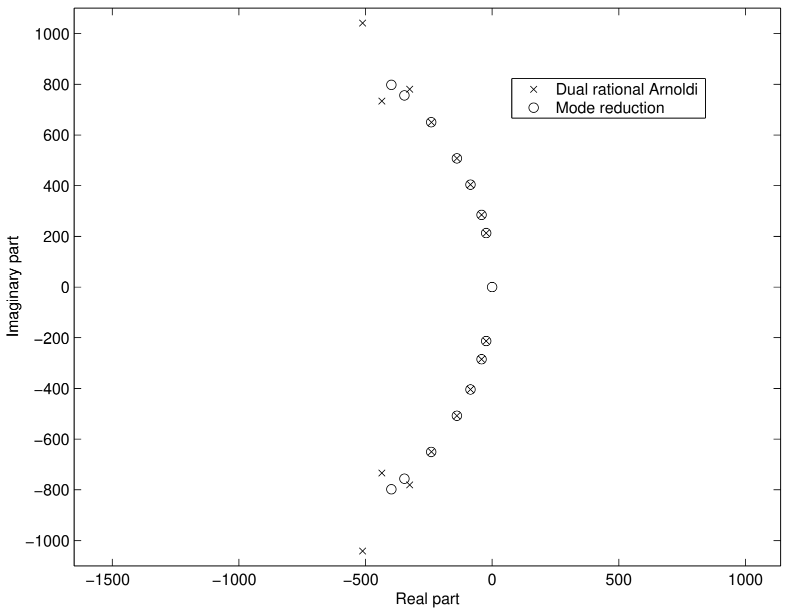


Figure 4.15: Eigenvalues of the reduced order pencils in the vibrating machines problem.

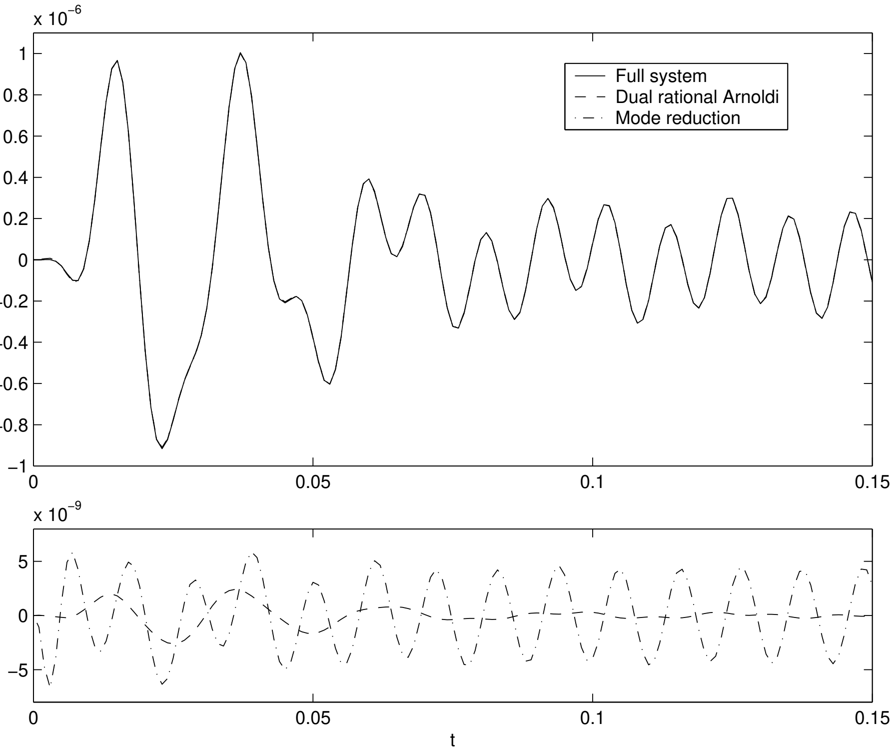


Figure 4.16: Output values $y(t)$ for the full order system, and $\widehat{y}(t)$ for the reduced order systems, in the vibrating machines problem. The lower graph shows the differences $y(t) - \widehat{y}(t)$ for dual rational Arnoldi (dash) and mode reduction (dashdot).

Chapter 5

Conclusions and Future Work

We have implemented a dual rational Arnoldi algorithm for model order reduction in FEMLAB and seen that for some problems the reduced order model is a good approximation. This method should be better than the mode reduction for non-normal systems and for input signals of high frequencies, and our numerical results indicate that this is the case.

There are a number of interesting subjects for future work that we want to conclude this present work by pointing out.

It is not clear how to place the shift points in an optimal way. With an accurate, but cheaply computed, estimate of the error in the approximation of the transfer function, an adaptive algorithm could be designed.

The use of an iterative solver for the shifted systems is not studied in detail here. We recommend the users to solve the systems to full accuracy, and it is important that they use a good preconditioner.

An implicit restarting procedure for stabilisation could be derived and compared with the method used here.

Finally, the method presented here uses a state space form of a linear system or a linearisation of a nonlinear system, and is not well suited to strongly nonlinear or time-varying systems. For nonlinear systems a method such as the proper orthogonal decomposition (POD) could be used. Willcox, Peraire and White [39] compare POD and Arnoldi methods for a turbomachinery problem. A multipoint Krylov method for time-varying systems is presented by Phillips [31].

Bibliography

[1] J. I. Aliaga, D. L. Boley, R. W. Freund, and V. Hernández. A Lanczos-type method for multiple starting vectors. *Mathematics of Computation*, 69(232):1577–1601, 2000.

[2] A. C. Antoulas and D. C. Sorensen. Approximation of large-scale dynamical systems: An overview. Technical report, Rice University, Houston, 2001.

[3] Z. Bai, P. Feldmann, and R. Freund. How to make theoretically passive reduced-order models passive in practice. In *Proceedings of the IEEE 1998 Custom Integrated Circuits Conference*, pages 207–210. IEEE, 1998.

[4] Z. Bai and R. W. Freund. A partial Padé-via-Lanczos method for reduced-order modeling. *Linear Algebra and its Applications*, 332-334:139–164, 2001.

[5] Z. Bai and Q. Ye. Error estimation of the Padé approximation of transfer functions via the Lanczos process. *Electronic Transactions on Numerical Analysis*, 7:1–17, 1998.

[6] J. H. Brandts. Matlab code for sorting real Schur forms. *Numerical Linear Algebra with Applications*, 9(3):249–261, 2002.

[7] C.-C. Chu and H.-J. Lee. Applications of Multi-Point Arnoldi Algorithms to Linear Lumped Transformer Simplifications. In *Power Engineering Society Summer Meeting*, volume 4, pages 2406–2411. IEEE, 2000.

[8] COMSOL AB. *FEMLAB Structural Mechanics Module*, First printing FEMLAB 2.3, 2002.

[9] COMSOL AB. *FEMLAB Reference Manual*, Second printing FEMLAB 2.2, 2001.

[10] J. Cullum and T. Zhang. Two-sided Arnoldi and nonsymmetric Lanczos algorithms. *SIAM Journal on Matrix Analysis and Applications*, 24(2):303–319, 2002.

[11] I. M. Elfadel and D. D. Ling. A block rational Arnoldi algorithm for multipoint passive model-order reduction of multiport RLC networks. In *Proceedings of the 1997 IEEE/ACM International Conference on Computer-Aided Design*, pages 66–71. IEEE Computer Society, Washington, DC, USA, 1997.

[12] K. Eriksson, D. Estep, P. Hansbo, and C. Johnson. *Computational Differential Equations*. Studentlitteratur, Lund, 1996.

[13] P. Feldmann and R. W. Freund. Efficient Linear Circuit Analysis by Padé Approximation via the Lanczos Process. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 14(5):639–649, 1995.

[14] R. W. Freund. Computation of Matrix Padé Approximations of Transfer Functions via a Lanczos-type Process. In C. K. Chui and L. L. Schumaker, editors, *Approximation Theory VIII*, volume 1: Approximation and Interpolation, pages 215–222. World Scientific Publishing Co., Singapore, 1995.

[15] R. W. Freund. Krylov-subspace methods for reduced-order modeling in circuit simulation. *Journal of Computational and Applied Mathematics*, 123:395–421, 2000.

[16] K. Gallivan, E. Grimme, and P. Van Dooren. Asymptotic waveform evaluation via a Lanczos method. *Applied Mathematics Letters*, 7:75–80, 1994.

[17] K. Gallivan, E. Grimme, and P. Van Dooren. A Rational Lanczos Algorithm for Model Reduction. *Numerical Algorithms*, 12:33–63, 1995.

[18] J. R. Gilbert, C. Moler, and R. Schreiber. Sparse matrices in MATLAB: Design and implementation. *SIAM Journal on Matrix Analysis and Applications*, 13(1):333–356, 1992.

[19] E. Grimme. *Krylov Projection Methods for Model Reduction*. PhD thesis, University of Illinois at Urbana-Champaign, 1997.

[20] E. Grimme and K. Gallivan. A Rational Lanczos Algorithm for Model Reduction II: Interpolation Point Selection. Technical report, University of Illinois at Urbana-Champaign, 1998.

[21] E. Grimme, K. A. Gallivan, V. de Clippel, and P. M. Van Dooren. Approximate Solves in Krylov-based Modeling Methods. In *Proceedings of the 36th Conference on Decision & Control, San Diego, California USA*, pages 3849–3854. IEEE, 1997.

[22] E. J. Grimme, D. C. Sorensen, and P. Van Dooren. Model reduction of state space systems via an implicitly restarted Lanczos method. *Numerical Algorithms*, 12:1–31, 1995.

[23] B. Kågström and P. Poromaa. Computing eigenspaces with specified eigenvalues of a regular matrix pair (A,B) and condition estimation: theory, algorithms and software. *Numerical Algorithms*, 12:369–407, 1996.

[24] B. Kågström and L. Westin. Generalized Schur Methods with Condition Estimators for Solving the Generalized Sylvester Equation. *IEEE Transactions on Automatic Control*, 34(7):745–751, 1989.

[25] I. M. Jaimoukha and E. M. Kasenally. Implicitly restarted Krylov subspace methods for stable partial realizations. *SIAM Journal on Matrix Analysis and Applications*, 18(3):633–652, 1997.

[26] R. B. Lehoucq and K. Meerbergen. Using generalized Cayley transformations within an inexact rational Krylov sequence method. *SIAM Journal on Matrix Analysis and Applications*, 20(1):131–148, 1998.

[27] J. Li. *Model Reduction of Large Linear Systems via Low Rank System Gramians*. PhD thesis, Massachusetts Institute of Technology, 2000.

[28] J. Li and J. White. Low rank solution of Lyapunov equations. *SIAM Journal on Matrix Analysis and Applications*, 24(1):260–280, 2002.

[29] T. V. Nguyen and J. Li. Multipoint Padé Approximation using a Rational Block Lanczos Algorithm. In *Proceedings of the 1997 IEEE/ACM International Conference on Computer-Aided Design*, pages 72–77. ACM and IEEE Computer Society, 1997.

[30] G. Obinata and B. D. O. Anderson. *Model Reduction for Control System Design*. Springer, New York, 2001.

[31] J. R. Phillips. Model Reduction of Time-Varying Linear Systems Using Approximate Multipoint Krylov-subspace Projectors. In *Digest of Technical Papers. 1998 IEEE/ACM International Conference on Computer-Aided Design*, pages 96–102. IEEE/ACM, 1998.

[32] A. Ruhe. The rational Krylov algorithm for nonsymmetric eigenvalue problems. III: Complex shifts for real matrices. *BIT*, 34:165–176, 1994.

[33] A. Ruhe and D. Skoogh. Rational Krylov algorithms for eigenvalue computation and model reduction. In B. Kågström, J. Dongarra, E. Elmroth, and J. Wasniewski, editors, *Applied Parallel Computing, Large Scale Scientific and Industrial Problems, 4th International Workshop, PARA '98, Umeå, Sweden, June 14–17, 1998, Proceedings*, volume 1541 of *Lecture Notes in Computer Science*, pages 491–502. Springer, 1998.

[34] L. F. Shampine and M. W. Reichelt. The MATLAB ODE Suite. *SIAM Journal on Scientific Computing*, 18(1):1–22, 1997.

[35] D. Skoogh. *Krylov Subspace Methods for Linear Systems, Eigenvalues and Model Order Reduction*. PhD thesis, Chalmers University of Technology, 1998.

[36] D. C. Sorensen and A. C. Antoulas. The Sylvester equation and approximate balanced reduction. *Linear Algebra and its Applications*, 351–352:671–700, 2002.

[37] T. Stykel. Model reduction of descriptor systems. Technical report, Institut für Mathematik, Technische Universität Berlin, 2001.

[38] D. S. Weile, E. Michielssen, and K. Gallivan. Reduced-Order Modeling of Multiscreen Frequency-Selective Surfaces Using Krylov-Based Rational Interpolation. *IEEE Transactions on Antennas and Propagation*, 49(5):801–813, 2001.

[39] K. Willcox, J. Peraire, and J. White. An Arnoldi approach for generation of reduced-order models of turbomachinery. *Computers & Fluids*, 31:369–389, 2002.