Term Project

Numerische Modellreduktion für komplexe Eigenwertprobleme

(Reduced Order Modelling for Complex Eigenvalue Problems)

Timur Aka
31.07.2003

Betreut durch

Prof. Dr. R. Hiptmair
Seminar for Applied Mathematics
Swiss Federal Institute of Technology
Zürich

Inhaltsverzeichnis

1 Einleitung

Gegeben sei das komplexe Eigenwertproblem der Form

$$\text{-div}\left(\alpha\left(x\right)\ \text{grad}\ u\right) = \omega^2 u \qquad \text{in } \Omega,$$

mit absorbierenden Randbedingungen auf $\partial\Omega$.

Die Diskretisierung des Problems erfolgt durch lineare finite Elemente auf einem regulären Gitter. Die absorbierenden Randbedingungen werden durch sogenannte PML (perfectly matched layers) realisiert. Die Berechnung der Eigenwerte und Eigenfunktionen erweist sich als schwierig, da wegen der PML die Steifigkeits- und Massenmatrix komplexe Elemente beinhalten.

Die Lösungsstrategie besteht darin, anstelle der PML zunächst Neumann Randbedingungen zu erzwingen und einige kleine Eigenmoden des reellen EWP zu berechnen. In einem nächsten Schritt werden diese Eigenmoden als Basisfunktionen zusammen mit gewöhnlichen finiten Elementen in der PML-Schicht zur Diskretisierung des komplexen EWP benutzt. Dieser Ansatz wird "Reduced Order Modelling" genannt.

Die Methode wird in MATLAB implementiert. Die Genauigkeit der Methode soll in Abhängigkeit der Anzahl benutzter Eigenmoden untersucht werden.

2 Formulierung

Wir betrachten das Eigenwertproblem in 1d. Die Ausdehnung der PML-Schicht wird durch t vorgegeben.

2.1 Das stetige Modellproblem

Für $t \geq 0$ sei Ω das Intervall $(-t, 1+t)$. Wir betrachten folgendes Variations-Eigenwertproblem: Finde $u \in H^1(\Omega)$, $\omega \in \mathbb{C}$, so dass

$$\left(\mu^{-1}\nabla u, \nabla v\right)_{L^2(\Omega)} = \omega^2 \left(\epsilon u, v\right)_{L^2(\Omega)} \qquad \forall v \in H^1(\Omega)$$

Die Koëffizienten $\mu, \epsilon \in L^\infty(\Omega)$ sind in $\widetilde{\Omega} = (0,1)$ reell und positiv, in $\Omega \backslash [0,1]$ hingegen sind sie zur Realisierung der absorbierenden Randbedingungen (PML) komplex.

$$\mu(x) = \begin{cases} \frac{1}{1+i\alpha\left(\frac{x}{t}\right)^2} & \text{für} \quad x < 0 \\ \mu_0\left(x\right) & \text{für} \quad 0 < x < 1 \\ \frac{1}{1-i\alpha\left(\frac{x-1}{t}\right)^2} & \text{für} \quad x > 1 \end{cases}$$

$$\epsilon(x) = \begin{cases} \frac{1}{1+i\alpha\left(\frac{x}{t}\right)^2} & \text{für} \quad x < 0 \\ \epsilon_0\left(x\right) & \text{für} \quad 0 < x < 1 \\ \frac{1}{1-i\alpha\left(\frac{x-1}{t}\right)^2} & \text{für} \quad x > 1 \end{cases}$$

α ist die Dämpfungskonstante. Es wird im Allgemeinen $\alpha \approx 5$ gewählt.

2.2 Das diskrete Modellproblem

Wir verwenden eine Galerkin-Diskretisierung mit stückweise linearen Basisfunktionen (Lagrange-Polynome 1. Ordnung) $b_j(x)$ auf einem uniformen Gitter mit Maschenweite h in Ω.

$$\mathcal{T}_h := \{[x_{i-1}, x_i]\}_{i=0}^{N-1}, \quad x_i = -t + i \cdot h, \quad h := \tfrac{1+2t}{N}$$

$$b_j(x) = \begin{cases} \frac{1-|x-x_j|}{h} & x_{j-1} \le x \le x_{j+1} \\ 0 & sonst \end{cases}$$

Zu beachten ist, dass t und 1 ganzzahlige Vielfache von h sind, so dass sich die Stützstellen 0 und 1 mit Gitterpunkten decken.

Auf diese Weise erhalten wir das diskrete Eigenwertproblem

$$A_h \underline{u} = \omega^2 M_h \underline{u}\,,$$

wobei $A_h \in \mathbb{C}^{N+1,N+1}$ die Steifigkeitsmatrix, $M_h \in \mathbb{C}^{N+1,N+1}$ die Massenmatrix und $\underline{u}$ der Vektor mit den approximierten Knotenwerten an den Gitterpunkten ist. Sowohl die Massen- als auch die Steifigkeitsmatrix sind tridiagonal und werden aus Elementmatrizen assembliert. Im Intervall (x_i, x_{i+1}) ist die Elementsteifigkeitsmatrix von der Form

$$A_i = \frac{1}{h^2} \int_{x_i}^{x_{i+1}} \mu^{-1}(x)\, dx \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

und die Elementmassenmatrix

$$M_i = \int_{x_i}^{x_{i+1}} \epsilon(x) \begin{pmatrix} \left(1 - \frac{x-x_j}{h}\right)^2 & (1 - \frac{x-x_j}{h})(\frac{x-x_j}{h}) \\ (\frac{x-x_j}{h})(1 - \frac{x-x_j}{h}) & \left(\frac{x-x_j}{h}\right)^2 \end{pmatrix} dx$$

2.3 Reduced Order Modelling

Anstatt das vollständig diskretisierte, komplexe Eigenwertproblem direkt zu lösen, wird in einem ersten Schritt das reelle EWP im Bereich $\widetilde{\Omega} = (0,1)$ mit Neumann Randbedingungen diskretisiert. Von diesem Matrixeigenwertproblem wird eine bestimmte Anzahl k der kleinsten Eigenpaare (λ_i, v_i) berechnet. Die so erhaltenen Eigenfunktionen liefern zusammen mit gewöhnlichen finiten Elementen in der PML-Schicht die Basisfunktionen des komplexen EWP in $\Omega = (-t, 1+t)$.

Sei k der Grad des Reduced Order Modelling, das heisst die Anzahl zu berechnenden Eigenpaare des Neumann-EWP. Bei $k = \frac{1}{h} + 1 = L + 1$ entspricht das Reduced Order Modelling einer vollständigen Diskretisierung. Das Ersetzen der stückweise linearen Funktionen in $\widetilde{\Omega} = (0,1)$ durch die Eigenmoden kommt einer Basistransformation gleich.

$$\mathcal{B}^{'} \overset{S}{\curvearrowright} \mathcal{B}$$

mit

$$\mathcal{B} = \{b_0, b_1, \ldots, b_N\}$$
$$\mathcal{B}^{'} = \{b_0, b_1, \ldots, b_{\frac{P}{2}-1}, v_1, \ldots, v_{L+1}, b_{\frac{P}{2}+L+1}, \ldots, b_{L+P}\}$$

wobei

$$b_0 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, b_1 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \ \ldots; N = \frac{1+2t}{h} = L + P$$

und v_i die Eigenmoden des Neumann-Eigenwertproblems sind. Dieser Umstand kann zur Überprüfung der Implementierung benutzt werden.

Es seien A_{full} und M_{full} die Steifigkeits- und Massenmatrix des vollständig diskretisierten EWP und A_{red}, M_{red} die entsprechenden Matrizen des Reduced Order Modelling k-ten Grades mit $k = L + 1 = \frac{1}{h} + 1$. Sei S die Transformationsmatrix und habe die Form

$$S = \begin{pmatrix} I & & 0 \\ & V & \\ 0 & & I \end{pmatrix}.$$

I ist die Einheitsmatrix und V beinhaltet spaltenweise die Werte der Eigenfunktionen des Eigenwertproblems ohne PML an den Gitterpunkten. So muss gelten:

$$\begin{aligned} A_{full}x &= \lambda M_{full}x \\ S^T A_{full} S x &= \lambda S^T M_{full} S x \\ A_{red}x &= \lambda M_{red}x \end{aligned}$$

$$\Rightarrow A_{red} = S^T A_{full} S \text{ und } M_{red} = S^T M_{full} S$$

3 Implementierung (1d)

3.1 Implementierung elementarer Funktionen

Wie erwähnt verwenden wir zur Diskretisierung sogenannte Hutfunktionen $b_j(x)$ als Basen. Da wir die Steifigkeits- und Massenmatrix aus Elementmatrizen assemblieren möchten, bedarf es der Unterscheidung

$$b_j(x) = \begin{cases} {}^1b_j(x) = 1 - \frac{x - x_j}{h} & x_j \leq x \leq x_{j+1} \\ {}^2b_j(x) = \frac{x - x_j}{h} & x_{j-1} \leq x \leq x_j \end{cases}$$

Implementiert sind die Hutfunktionen in `base1()` und `base2()`.

```
% Input: x, die Stelle, an welcher die Funktion ausgewertet werden soll
%        xl, linker Rand des Intervalls
%        h, Schrittweite
% Output: y=b(x)
function [y]=base1(x,xl,h)
y=1-(x-xl)/h;
return

function [y]=base2(x,xl,h)
y=(x-xl)/h;
return
```

Des Weiteren haben wir die Funktionen $\mu(x)$ und $\epsilon(x)$, welche die absorbierenden Randbedingungen realisieren.

```
% Input:  x, auszuwertende Stelle
%         t, Betrag der PML-Schicht
%         alpha, Kontrollparameter
% Output: y=mu(x) bzw epsilon(x)
function [y]=mu(x,t,alpha)
% der Einfachheit halber sei mu0(x)=konst=1
if (x>=0 & x<=1)
   y=1;
end
if (x>1)
   y=1/(1-alpha*i*((x-1)/t)^2);
end
if (x<0)
   y=1/(1+alpha*i*(x/t)^2);
end
return

function [y]=epsilon(x,t,alpha)
% der Einfachheit halber sei epsilon0(x)=konst=1
if (x>=0 & x<=1)
   y=1;
end
if (x>1)
   y=1/(1-alpha*i*((x-1)/t)^2);
end
if (x<0)
   y=1/(1+alpha*i*(x/t)^2);
end
return
```

Zur Berechnung der Elementsteifigkeits- bzw. der Elementmassenmatrizen müssen Polynome vom Grad 3 bzw. 5 numerisch integriert werden. Dazu verwenden wir

jeweils eine p-Punkt Quadratur-Formel, welche für Polynome (2p-1)-ten Grades im Intervall (-1,1) exakt ist.

```
% Input:  p, p-Punkt-Quadratur, exakt bis Polynom (2p-1)-ten Grades
% Output: x, Nullstellen dh Stuetzstellen
%         w, Gewichte
function [x,w]=gaussQuad(p)
x=zeros(p,1);
w=zeros(p,1);
b=zeros(p-1,1);
for i=1:(p-1)
   b(i)=i/sqrt(4*i*i-1);
end
J=diag(b,-1)+diag(b,1);
[ev, ew]=eig(J);
for i=1:p
   r=norm(ev(:,i));
   ev(:,i)./r;
end
x=diag(ew);
w=(2*(ev(1,:).*ev(1,:)))';
return
```

Für die Integration des Polynomes $f(x)$ vom Grad n über ein beliebiges Gebiet $[a, b]$ sieht die Verwendung der $(\frac{n+1}{2})$ Stützstellen x_i und Gewichte ω_i folgendermassen aus:

$$\int_a^b f(x)dx = m \cdot \sum_{i=1}^{\frac{n+1}{2}} f(mx_i + q) \cdot \omega_i \quad \text{mit} \quad m = \frac{b-a}{2} \quad \text{und} \quad q = \frac{a+b}{2}$$

3.2 Steifigkeits- und Massenmatrix des Neumann-EWP

Sowohl die Massen- als auch die Steifigkeitsmatrix werden aus Elementmatrizen assembliert. Die Elementsteifigkeitsmatrix hat im Intervall (x_i, x_{i+1}) die Form

$$\begin{aligned} A_i &= \int_{x_i}^{x_{i+1}} \mu^{-1}(x) \begin{pmatrix} \nabla({}^1b_i)\nabla({}^1b_i) & \nabla({}^1b_i)\nabla({}^2b_{i+1}) \\ \nabla({}^2b_{i+1})\nabla({}^1b_i) & \nabla({}^2b_{i+1})\nabla({}^2b_{i+1}) \end{pmatrix} dx \\ &= \frac{1}{h^2} \int_{x_i}^{x_{i+1}} \mu^{-1}(x)\, dx \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \end{aligned}$$

und die Elementmassenmatrix

$$\begin{aligned} M_i &= \int_{x_i}^{x_{i+1}} \epsilon(x) \begin{pmatrix} {}^1b_i\,{}^1b_i & {}^1b_i\,{}^2b_{i+1} \\ {}^2b_{i+1}\,{}^1b_i & {}^2b_{i+1}\,{}^2b_{i+1} \end{pmatrix} dx \\ &= \int_{x_i}^{x_{i+1}} \epsilon(x) \begin{pmatrix} \left(1 - \frac{x-x_j}{h}\right)^2 & (1 - \frac{x-x_j}{h})(\frac{x-x_j}{h}) \\ (\frac{x-x_j}{h})(1 - \frac{x-x_j}{h}) & \left(\frac{x-x_j}{h}\right)^2 \end{pmatrix} dx \end{aligned}$$

In den Funktionen `elemStiff(xl,h,p,t,alpha)` bzw. `elemMass(xl,h,p,t,alpha)` ist die Berechnung der Elementmatrizen umgesetzt, wobei `xl` für den linken Rand des Intervalls $K_i = (x_i, x_{i+1})$ und `h` für die Schrittweite $|K_i|$ steht. Das Argument `p` bestimmt die Quadratur-Formel, so dass die Approximation der Integrale für Polynome (2p-1)-ten Grades exakt ist. Des Weiteren ist `t` der Betrag der PML und `alpha` ein Kontrollparameter.

```
% Input:  xl, linke Intervallsgrenze
%         h, Schrittweite
%         p, p-Punkt-Quadratur
%         t, Betrag der PML-Schicht
%         alpha, Kontrollparameter
% Output: A_h, Elementsteifigkeits bzw. ElementMassenmatrix
%              im Intervall [xl,xl+h]
function [A_h]=elemStiff(xl,h,p,t,alpha)
A_h=zeros(2,2);
a=xl;
b=xl+h;
m=(b-a)/2;
q=(b+a)/2;
[x,w]=gaussQuad(p);
for i=1:length(x)
   mu=mu(m*x(i)+q,t,alpha);
   A_h=A_h+w(i)*1/mu;
end A_h=m*A_h; A_h=1/(h*h)*A_h*[1 -1;-1 1];
return

function [M_h]=elemMass(xl,h,p,t,alpha)
M_h=zeros(2,2); a=xl;
b=xl+h; m=(b-a)/2; q=(b+a)/2; [x,w]=gaussQuad(p);
for i=1:length(x)
    b1=base1(m*x(i)+q,a,h);
    b2=base2(m*x(i)+q,a,h);
    eps=epsilon(m*x(i)+q,t,alpha);
    M_h(1,1)=M_h(1,1)+w(i)*eps*b1*b1;
    M_h(1,2)=M_h(1,2)+w(i)*eps*b1*b2;
    M_h(2,1)=M_h(2,1)+w(i)*eps*b2*b1;
    M_h(2,2)=M_h(2,2)+w(i)*eps*b2*b2;
end M_h=m*M_h;
return
```

Die Assemblierung der globalen Matrizen geschieht mit Hilfe von sogenannten T-Matrizen, welche die Position der Elementmatrizen im globalen Linearsystem festlegen.

Beispielsweise wird die Elementsteifigkeitsmatrix A_i durch die Multiplikation $T_i^T A_i T_i$ in die richtige Form gebracht und anschliessend erhält man durch Summation aller $\{T_i^T A_i T_i\}_{i=1}^N$ die globale Steifigkeitsmatrix, wobei die T-Matrizen folgende Gestalt haben

$$T_i^T = \begin{pmatrix} 0 & 0 \\ \vdots & \vdots \\ 1 & 0 \\ 0 & 1 \\ \vdots & \vdots \\ 0 & 0 \end{pmatrix} \begin{matrix} \\ \\ i \\ i{+}1 \\ \\ \end{matrix} \begin{matrix} 1 \\ \vdots \\ \\ \\ \vdots \\ N{+}1 \end{matrix}$$

3.3 Lösen des Neumann-EWP

In einem ersten Schritt möchten wir das gegebene Eigenwertproblem für das Gebiet $\widetilde{\Omega} = (0, 1)$ berechnen, in welchem die Funktionswerte von $\mu(x)$ und $\epsilon(x)$ gemäss Aufgabenstellung reell und positiv sind. Es sei gegeben ein uniformes Gitter:

$$\mathcal{T}_h = \{[x_{j-1}, x_j]\}_{j=1}^L \qquad x_j = j \cdot h \qquad h = \frac{1}{L}$$

Zu beachten ist, dass t und 1 ganzzahlige Vielfache von h sind, so dass sich bei späterem Einbezug der PML, der Rand des Gebietes $\widetilde{\Omega}$, also die Stützstellen 0 und 1, mit Gitterpunkten deckt.

Die Assemblierung der globalen Matrizen geling durch eine einfache **for**-Schlaufe über die Teilintervalle $\{K_j\}_{j=1}^L$.

```
% Input:  h, Schrittweite
%         t, Betrag der PML-Schicht
%         alpha, Kontrollparameter
% Output: V, Matrix, die spaltenweise Eigenvektoren des Neumann-EWP enthaltet
%         D, Diagonalmatrix mit Eigenwerten zu V
function [V,D]=ewp_Neumann(h,t,alpha)
format long;
N=1/h;
% ************************************
% *Loesen des Eigenwertproblems ohne PML*
% ************************************
% A, globale Steifigkeitsmatrix
% M, globale Massenmatrix
% T, T-Matrizen zur Assemblierung
A=zeros(N+1);
M=zeros(N+1);
for i=1:(N)
    T=zeros(N+1,2);
    T(i,1)=1;
    T(i+1,2)=1;
```

```
    A=A+T*elemStiff((i-1)*h,h,2,t,alpha)*T';
    M=M+T*elemMass((i-1)*h,h,3,t,alpha)*T';
end
```

Das so erhaltene Eigenwertproblem

$$A\underline{u} = \omega^2 M\underline{u} \text{ , mit } dim(A) = dim(M) = N + 1$$

kann mit der MATLAB-Funktion `eig()` gelöst werden, welche uns $L + 1$ Eigenpaare liefert.

```
[V,D]=eig(A,M);
% Normierung der Eigenvektoren
for i=1:N+1
    V(:,i)=V(:,i)./norm(V(:,i));
end return
```

3.4 Basisfunktionen für Spektral-Galerkin

Gemäss Reduced Order Modelling werden zur Diskretisierung des komplexen Eigenwertproblems eine bestimmte Anzahl k der Eigenfunktionen aus dem Neumann-EWP hinzugezogen. Die zugehörigen reellen Eigenwerte dieser Eigenfunktionen sollen in der Nähe eines kritischen Wertes $\lambda_{krit} \in \mathbb{R}$ liegen.

Die Funktion `getBase()` liefert eine Matrix, welche spaltenweise die k gewünschten diskreten Eigenmoden enthält. Es seien $(\lambda_i, v_i)_{i=1}^{L+1}$ die Eigenpaare des Neumann-EWP, λ_{krit} und die Dimension k der zu bildenden Basis gegeben. Die relevanten Eigenmoden werden wie folgt ermittelt:

inc, dec, count=0

1. falls $k > count$
   - wähle Eigenpaar $(\lambda_i, v_i)_{i=0}^{L+1}$, so dass
     $$|\lambda_{krit} - \lambda_i| = min$$
   - $\hookrightarrow v_i$ (Basisfunktion)
   - *count++, inc=dec=i*
   - falls $k \leq count$ $\rightarrow$ fertig
2. falls $dec > 1$
   - *dec--*
   - $\hookrightarrow v_{dec}$ (Basisfunktion)
   - *count++*
   - falls $k \leq count$ $\rightarrow$ fertig
3. falls $inc < L + 1$
   - *inc++*
   - $\hookrightarrow v_{inc}$ (Basisfunktion)

- $count{+}{+}$
- falls $k \leq count \rightarrow$ fertig

4. wiederhole Schritte 2. bis 4.

Es sei erwähnt, dass $k \leq L + 1$ sein muss.

```
% Input:  V, Matrix, die spaltenweise Eigenvektoren enthaltet
%         D, Diagonalmatrix mit Eigenwerten zu V
%         r0, kritischer Wert
%         nrOfEM, Grad des REDUCED ORDER MODELLING
% Output: Base, Matrix, die spaltenweise Basisfunktionen des Reduced
%         Order Modelling enthaltet
function [Base]=getBase(V,D,r0,nrOfEM)
Base=zeros(size(V,1),nrOfEM);
D_list=D*ones(size(D,2),1);
[sorted,mapping]=sort(D_list);
indexBT=mapping(find(sorted>r0));
indexSE=mapping(find(sorted<=r0));
index=zeros(nrOfEM,1);
maxIndex=max(size(indexBT,1),size(indexSE,1));
m=1;

for i=1:maxIndex
    if i<=size(indexSE,1)
        index(m,1)=indexSE(size(indexSE,1)-i+1);
        m=m+1;
    end
    if i<=size(indexBT,1)
        index(m,1)=indexBT(i);
        m=m+1;
    end
end for j=1:nrOfEM
    Base(:,j)=V(:,index(j));
end return
```

3.5 Diskretisierung des PML-EWP mit Reduced Order Modelling

Für das Eigenwertproblem mit Dirichlet-Randbedingungen im Gebiet $\Omega = (-t, 1 + t)$ sei der Lösungsraum $V = \{u \in H^1(\Omega)\,;\, u|_{\partial\Omega} = 0\}$. Die Variationsform lautet somit:

Finde $u \in H^1(\Omega)$, $\omega \in \mathbb{C}$, so dass

$$\left(\mu^{-1}\nabla u, \nabla v\right)_{L^2(\Omega)} = \omega^2 \left(\epsilon u, v\right)_{L^2(\Omega)} \qquad \forall v \in H^1(\Omega)$$

Zur Diskretisierung des Problems verwendet man nun zum einen finite Elemente im PML-Gebiet, zum anderen Spektral-Galerkin in $\widetilde{\Omega} = (0, 1)$. Der Grad des

Reduced Order Modelling sei gegeben $k \leq \frac{1}{h} + 1 = L + 1$.

$$u = \sum_{i=1}^{k} \alpha_i e_i + \sum_{j=1}^{P} \beta_j b_j$$

Wobei b_j Hutfunktionen im PML-Gebiet und e_i Eigenfunktionen in $\widetilde{\Omega} = (0,1)$ sind. Testen dieser Linearform mit $v = e_K$ bzw. $v = b_K$ liefert uns

$$\textstyle\sum_i \alpha_i(\mu^{-1}\nabla e_i, \nabla e_K) + \sum_j \beta_j(\mu^{-1}\nabla b_j, \nabla e_K) = \omega^2 \left(\sum_i \alpha_i(\epsilon\, e_i, e_K) + \sum_j \beta_j(\epsilon\, b_j, e_K)\right)$$

$$\forall K = 1 \ldots k$$

$$\textstyle\sum_i \alpha_i(\mu^{-1}\nabla e_i, \nabla b_K) + \sum_j \beta_j(\mu^{-1}\nabla b_j, \nabla b_K) = \omega^2 \left(\sum_i \alpha_i(\epsilon\, e_i, b_K) + \sum_j \beta_j(\epsilon\, b_j, b_K)\right)$$

$$\forall K = 1 \ldots P$$

Die Steifigkeitsmatrix A bzw. Massenmatrix M bekommt so die Form:

$$\begin{pmatrix} \begin{pmatrix} e_i \Leftrightarrow e_K \\ \texttt{(I)} \end{pmatrix} & \begin{pmatrix} e_i \Leftrightarrow b_K \\ \texttt{(II)} \end{pmatrix} \\ \begin{pmatrix} b_i \Leftrightarrow e_K \\ \texttt{(III)} \end{pmatrix} & \begin{pmatrix} b_i \Leftrightarrow b_K \\ \texttt{(IV)} \end{pmatrix} \end{pmatrix}$$

Die Dimension dieser Matrix ergibt sich aus dem Grad des Reduced Order Modelling und der Zerlegung des PML-Gebietes in Teilintervalle. Bei uniformem Gitter über das ganze Gebiet Ω bekommt man in den PML-Schichten $(-t, 0)$ und $(1, 1+t)$ je $P = \frac{|t|}{h}$ Teilintervalle und somit $P+1$ Basen. Aus Gründen, die später genauer erläutert werden, verwenden wir aber nur je P Basen aus der PML-Schicht. Die Matrixpartition `(I)` hat die Dimension $(k) \times (k)$, die Partition `(II)` $(k) \times (2P)$, `(III)` $(2P) \times (k)$ und `(IV)` $(2P) \times (2P)$. Für die gesammte Matrix ergibt das $dim(A) = dim(M) = (k + 2P)^2$.

Die Elemente der Matrixpartition `(I)` ergeben sich für die Steifigkeitsmatrix durch Testen der k Eigenfunktionen des Neumann-EWP gegeneinander. Durch erneutes zerlegen der Eigenfunktionen in Linearkombinationen aus Hutfunktionen $e_i = \sum_{l=1}^{N+1} \kappa_l^{(i)} b_l$, wobei $\kappa_l^{(i)}$ den Werten der Eigenfunktion e_i an den Stützstellen $\{x_l\}_{l=1}^{N+1}$ im Gebiet $\widetilde{\Omega}$ entspricht, erhält man für $i, j = 1 \ldots k$

$$A_{i,j}^{(I)} \quad = \quad \left(\mu^{-1}\nabla e_i, \nabla e_j\right)_{L^2(\Omega)}$$

$$
\begin{aligned}
&= \left(\mu^{-1}\nabla \sum_{l=1}^{N+1} \kappa_l^{(i)} b_l, \nabla \sum_{m=1}^{N+1} \kappa_m^{(j)} b_m\right)_{L^2(\Omega)} \\
&= (\kappa^{(i)})^T S \kappa^{(j)}
\end{aligned}
$$

Die Matrix S ist tridiagonal und entspricht im Grunde der globalen Steifigkeitsmatrix in $\widetilde{\Omega}$, mit dem Unterschied, dass die Basen (Hutfunktionen), um die Stetigkeit der Eigenfunktionen am Rand von $\widetilde{\Omega}$ zu gewährleisten, im Intervall $(-h, 0)$ und $(1, 1+h)$ stetig nach 0 abfallend forgesetzt werden müssen. Dies hat lediglich auf die Matrixelemente $S_{1,1}$ und $S_{N+1,N+1}$ Einfluss.

$$
\begin{aligned}
S_{1,1} &= \left(\mu^{-1}\nabla b_1, \nabla b_1\right) \\
&= \int_{-h}^{h} \mu^{-1}\nabla b_1 \nabla b_1 dx \\
&= \int_{-h}^{0} \mu^{-1}\nabla^2 b_1 \nabla^2 b_1 dx + \int_{0}^{h} \mu^{-1}\nabla^1 b_1 \nabla^1 b_1 dx
\end{aligned}
$$

Da das Matrixelement $S_{1,1}$ nach der Assemblierung schon den Wert des Integrals $\int_0^h \ldots dx$ hat, genügt es, das Element mit $\int_{-h}^0 \ldots dx$ zu addieren. Analoges gilt für $S_{N+1,N+1}$.

```
% Input:  h, Schrittweite
%         t, Betrag der PML-Schicht
%         alpha, Kontrollparameter
%         Base, Matrix, die spaltenweise Basisfunktionen des Reduced
%               Order Modelling enthaltet
% Output: V, Matrix, die spaltenweise Eigenvektoren des PML-EWP enthaltet
%         D, Diagonalmatrix mit Eigenwerten zu V
function [V,D]=ewp_RedOrdMod(h,t,alpha,Base)
format long;
N=1/h;
P=t/h;
% *****************************************
% *Zusammensetzen der Matrizen A und M mit PML*
% *****************************************
% A_ee, Matrixpartition e(i) <-> e(j) der Steifigkeitsmatrix
% M_ee, Matrixpartition e(i) <-> e(j) der Massenmatrix
% A_temp, Steifigkeitsmatrix ohne PML
% M_temp, Massenmatrix ohne PML
A_ee=zeros(N+1);
M_ee=zeros(N+1);
A_temp=zeros(N+1);
M_temp=zeros(N+1);
```

```
for i=1:(N)
    T=zeros(N+1,2);
    T(i,1)=1;
    T(i+1,2)=1;
    A_temp=A_temp+T*elemStiff((i-1)*h,h,2,t,alpha)*T';
    M_temp=M_temp+T*elemMass((i-1)*h,h,3,t,alpha)*T';
end
% Berechnen der Integrale im Intervall (-h,0) und (1,1+h) und Addieren
% zu den Matrixelementen an erster und letzter Stelle
p_A=2;
p_M=3;
A_left=elemStiff(-h,h,p_A,t,alpha);
A_right=elemStiff(1,h,p_A,t,alpha);
A_temp(1,1)=A_temp(1,1)+A_left(2,2);
A_temp(N+1,N+1)=A_temp(N+1,N+1)+A_right(1,1);
M_left=elemMass(-h,h,p_M,t,alpha);
M_right=elemMass(1,h,p_M,t,alpha);
M_temp(1,1)=M_temp(1,1)+M_left(2,2);
M_temp(N+1,N+1)=M_temp(N+1,N+1)+M_right(1,1);
% Multiplizieren mit den Werten der Eigenfunktion an den Stuetzstellen
A_ee=conj(Base')*A_temp*Base; M_ee=conj(Base')*M_temp*Base;
```

Die Elemente der Partition `(II)` $A^{(II)}_{i,j}$ für $i = 1 \dots k$, $j = 1 \dots 2P$ ergeben sich aus $(\mu^{-1}\nabla e_i, \nabla b_j)$. Wir nummerieren die Hutfunktionen in der PML Schicht in den Gebieten $(-t, 0)$ und $(1, 1 + t)$ von links her beginnend durch. Somit ist

$$\begin{array}{rll} b_1(x) \neq 0 & \text{für} & -t < x < -t + h \\ b_2(x) \neq 0 & \text{für} & -t < x < -t + 2h \\ \vdots & & \\ b_P(x) \neq 0 & \text{für} & -2h < x < 0 \\ b_{P+1}(x) \neq 0 & \text{für} & 1 < x < 1 + 2h \\ \vdots & & \\ b_{2P}(x) \neq 0 & \text{für} & 1 + t - h < x < 1 + t \end{array}$$

Die Hutfunktionen $\widetilde{b}_1$ im Intervall $(-h, h)$ und $\widetilde{b}_{N+1}$ in $(1 - h, 1 + h)$ werden von den Eigenfunktionen aus $\widetilde{\Omega}$ bestimmt und ihre Koëffizienten entsprechen den Werten der Eigenfunktionen bei $x = 0$ und $x = 1$. Somit ist klar, dass für die Elemente der Partition `(II)` gilt:

$$A^{(II)}_{i,j} = 0, \text{ falls } j \neq (P, P + 1)$$

Daraus folgt:

$$\left(\mu^{-1}\nabla e_i, \nabla b_P\right) \quad = \quad \left(\mu^{-1}\nabla \sum_{l=1}^{N+1} \kappa_l^{(i)} b_l, \nabla b_P\right)$$

$$
\begin{aligned}
&= \left(\mu^{-1}\nabla\kappa_1^{(i)}\widetilde{b}_1, \nabla b_P\right) \\
&= \kappa_1^{(i)}\left(\mu^{-1}\nabla\widetilde{b}_1, \nabla b_P\right)
\end{aligned}
$$

$$
\begin{aligned}
\left(\mu^{-1}\nabla e_i, \nabla b_{P+1}\right) &= \left(\mu^{-1}\nabla\sum_{l=1}^{N+1}\kappa_l^{(i)}b_l, \nabla b_{P+1}\right) \\
&= \left(\mu^{-1}\nabla\kappa_{N+1}^{(i)}\widetilde{b}_{N+1}, \nabla b_{P+1}\right) \\
&= \kappa_{N+1}^{(i)}\left(\mu^{-1}\nabla\widetilde{b}_{N+1}, \nabla b_{P+1}\right)
\end{aligned}
$$

Das Berechnen der Partition `(III)` ist trivial, da $A^{(III)} = (A^{(II)})^T$. Die Partitionen `(II)` und `(III)` der Massenmatrix bekommt man analog.

```
% A_eb, Matrixpartition e(i) <-> b(j) der Steifigkeitsmatrix
% M_eb, Matrixpartition e(i) <-> b(j) der Massenmatrix
A_eb=zeros(size(Base,2),2*P);
M_eb=zeros(size(Base,2),2*P);
p_A=2;
p_M=3;
A_left=elemStiff(-h,h,p_A,t,alpha);
A_right=elemStiff(1,h,p_A,t,alpha);
M_left=elemMass(-h,h,p_M,t,alpha);
M_right=elemMass(1,h,p_M,t,alpha);
for j=1:size(Base,2)
    A_eb(j,P)=Base(1,j)*A_left(2,1);
    A_eb(j,P+1)=Base(N+1,j)*A_right(1,2);
    M_eb(j,P)=Base(1,j)*M_left(2,1);
    M_eb(j,P+1)=Base(N+1,j)*M_right(1,2);
end
```

Zu guter Letzt bleibt die Partition `(IV)`, welche durch Testen der Hutfunktionen in der PML-Schicht gegeneinander zustandekommt. Man kann die Matrix $A^{(IV)}$ aufteilen in

$$
\begin{pmatrix}
 & & & 0 & \dots & 0 \\
 & \{A_{i,j}^{(IV)}\}_{i,j=1}^{P} & & \vdots & \ddots & \vdots \\
 & & & 0 & \dots & 0 \\
0 & \dots & 0 & & & \\
\vdots & \ddots & \vdots & & \{A_{i,j}^{(IV)}\}_{i,j=P+1}^{2P} & \\
0 & \dots & 0 & & &
\end{pmatrix}
$$

Die zwei Teile mit Einträgen ungleich 0 sind jeweils tridiagonal und ihre Assemblierung geschieht wie gehabt. Einzig zu beachten ist, dass in den Elementen $A_{P,P}^{(IV)}$ und $A_{P+1,P+1}^{(IV)}$ die Stetigkeitsbedingung nicht verletzt wird.

```
% A_bb, Matrixpartition b(i) <-> b(j) der Steifigkeitsmatrix
% M_bb, Matrixpartition b(i) <-> b(j) der Massenmatrix
% A_left, tridiagonale Matrix der linken PML-Schicht
% A_right, tridiagonale Matrix der rechten PML-Schicht
% M_left, tridiagonale Matrix der linken PML-Schicht
% M_right, tridiagonale Matrix der rechten PML-Schicht
A_bb=zeros(2*P,2*P);
M_bb=zeros(2*P,2*P);
A_left=zeros(P,P);
A_right=zeros(P,P);
M_left=zeros(P,P);
M_right=zeros(P,P);

for j=1:(P-1)
    T=zeros(P,2);
    T(j,1)=1;
    T(j+1,2)=1;
    A_left=A_left+T*elemStiff(-t+(j-1)*h,h,2,t,alpha)*T';
    M_left=M_left+T*elemMass(-t+(j-1)*h,h,3,t,alpha)*T';
    A_right=A_right+T*elemStiff(1+h+(j-1)*h,h,2,t,alpha)*T';
    M_right=M_right+T*elemMass(1+h+(j-1)*h,h,3,t,alpha)*T';
end
% Berechnen der Integrale im Intervall (-h,0) und (1,1+h) und Addieren
% zu den betroffenen Matrixelementen, um Stetigkeit zu gewaehrleisten
p_A=2;
p_M=3;
A_left_right=elemStiff(-h,h,p_A,t,alpha);
A_right_left=elemStiff(1,h,p_A,t,alpha);
A_left(P,P)=A_left(P,P)+A_left_right(1,1);
A_right(1,1)=A_right(1,1)+A_right_left(2,2);
M_left_right=elemMass(-h,h,p_M,t,alpha);
M_right_left=elemMass(1,h,p_M,t,alpha);
M_left(P,P)=M_left(P,P)+M_left_right(1,1);
M_right(1,1)=M_right(1,1)+M_right_left(2,2);
% Zusammensetzten der Matrizen A_bb und M_bb
A_bb=[A_left zeros(P,P); zeros(P,P) A_right];

M_bb=[M_left zeros(P,P); zeros(P,P) M_right];
```

Nun können sowohl Steifigkeitsmatrix als auch Massenmatrix zusammengesetzt werden. Auf beiden Matrizen müssen noch die Dirichlet Randbedingungen erzwungen werden. Lösen kann man das EWP wiederum mit der MATLAB-Funktion `eig()`.

```
% Dirichlet-Randbedingungen erzwingen
A_bb=A_bb(2:2*P-1,2:2*P-1);
M_bb=M_bb(2:2*P-1,2:2*P-1);
```

```
A_eb=A_eb(:,2:2*P-1);
M_eb=M_eb(:,2:2*P-1);
% ************************************
% *Loesen des kompletten EWP (mit PML)*
% ************************************
% A_withPML, globale Steifigkeitsmatrix des EWP ueber den ganzen Bereich (-t,t+1)
% M_withPML, globale Massenmatrix des EWP ueber den ganzen Bereich (-t,t+1)
A_withPML=zeros(N+1+2*P,N+1+2*P);
M_withPML=zeros(N+1+2*P,N+1+2*P);

A_withPML=[A_ee A_eb; conj(A_eb') A_bb];

M_withPML=[M_ee M_eb;conj(M_eb') M_bb];
[V,D]=eig(A_withPML,M_withPML);
return
```

3.6 main()-Funktion

Das numerische Lösen des komplexen PML-Eigenwertproblems auf $\Omega = (-t, 1+t)$ mit Reduced Order Modelling wird durch die Funktion `main()` koordiniert. Schrittweite h, Dicke der PML-Schicht t, der kritische Wert λ_{krit}, die Dämpfungskonstante α und der Grad des Reduced Order Modelling k sind zu setzen. $k = \frac{1}{h} + 1$ liefert die gleichen Eigenwerte wie das direkte Lösen des vollständig mit finiten Elementen diskretisierte Eigenwertproblem. Die Funktion gibt die Eigenwerte des Modellproblems aus, wobei die Anzahl $\frac{2t}{h} + k$ ist.

```
function main()
clear;
format long;
% ************************************
% *folgende Parameter sind zu setzen   *
% *h, Schrittweite                     *
% *t, Dicke der PML-Schicht            *
% *alpha, Kontrollparameter            *
% *r0, kritischer Wert                 *
% *k, Grad des Reduced Order Modelling *
% ************************************
% *Output: Eigenwerte                  *
% ************************************
t=0.1;
h=0.005;
r0=1000;
k=150;
% Wichtig!! k <= (1/h)+1
alpha=5;
```

```
[V_neumann,D_neumann]=ewp_Neumann(h,t,alpha);
[Base]=getBase(V_neumann,D_neumann,r0,k);
[V_pml,D_pml]=ewp_RedOrdMod(h,t,alpha,Base);

D_pml*ones(size(D_pml,1),1)
return
```

4 Auswertung und Resultate

4.1 Rahmenbedingungen

In erster Linie interessieren wir uns für einige wenige Eigenwerte des Modellproblems in einem bestimmten kritischen Bereich $\lambda_{krit} \pm \delta_0 \lambda_{krit}$ mit $\delta_0 = [0, 1]$.

4.1.1 Wahl des kritischen Wertes

Gemäss dem vorgestellten Verfahren ist λ_{krit} entscheidend bei der Wahl der k diskreten Eigenmoden aus dem Neumann-EWP. Die k Eigenmoden, deren Eigenwerte am nächsten an λ_{krit} liegen, bilden die Basis der spektralen Galerkin-Diskretisierung. Es ist daher zu beachten, dass die approximierten Eigenpaare hinreichend nahe an den exakten Eigenpaaren des Neumann-EWP liegen. Einerseits wird Konvergenz durch Gitterverfeinerung erreicht, andererseits muss der Tatsache, dass kleine Eigenwerte schneller konvergieren, Rechnung getragen werden [Abb. (1)].

Dies impliziert, dass das Verfahren für kleine λ_{krit} bessere Resultate liefert, da die Eigenpaare, insbesondere die Eigenmoden, welche im Reduced Order Modelling die Basisfunktionen darstellen, weniger fehlerbehaftet sind.

Tabelle (1) zeigt den relativen Fehler $r(\lambda_{krit}, \widetilde{\lambda}) = \frac{|\lambda_{krit} - \widetilde{\lambda}|}{|\lambda_{krit}|}$ der Approximation $\widetilde{\lambda}$ zum exakten Eigenwert λ_{krit} in Abhängigkeit der Anzahl verwendeten Eigenmoden k.

4.1.2 Wahl des kritischen Bereiches

In sämtlichen Berechnungen wird $\delta_0 = [0, 1]$ so gewählt, dass der Realteil von genau drei Eigenwerte z_1, z_2, z_3 des vollständig mit finiten Elementen diskretisierten, komplexen EWP im kritischen Bereich liegen.

4.1.3 Fehlermass

Um die Güte der Approximation zu messen, wurde das Fehlermass r_{tot} eingeführt. Es ist die Summe der quadrierten Differenzenbeträge aus exakten Eigenwerten $\{z_i\}_{i=1}^3$ und den mit Reduced Order Modelling ermittelten Werten $\{y_i\}_{i=1}^3$.

$$r_{tot} = \frac{\sum_{i=1}^3 |z_i - y_i|^2}{\sum_{i=1}^3 |z_i|}$$

Tabelle 1: relativer Fehler der Approximation mit $h = 0.01$, $t = 0.1$, $\alpha = 5$

i	λ_i^{ex}	$r(\cdot), k = 10$	$r(\cdot), k = 20$	$r(\cdot), k = 50$	$r(\cdot), k = 75$	$r(\cdot), k = 100$
1	8.2	0.0864	0.0385	0.0120	0.0048	0.0002
2	32.8	0.0764	0.0355	0.0112	0.0048	0.0000
3	73.9	0.0855	0.0358	0.0108	0.0043	0.0002
4	131.4	0.0724	0.0306	0.0093	0.0040	0.0000
5	205.5	0.0814	0.0277	0.0079	0.0031	0.0001
6	296.1	0.0467	0.0173	0.0051	0.0022	0.0000
7	403.3	0.0187	0.0068	0.0021	0.0008	0.0001
8	527.3	0.1015	0.0130	0.0022	0.0007	0.0001
9	668.1	0.0856	0.0274	0.0064	0.0026	0.0000
10	825.7	0.0900	0.0396	0.0111	0.0042	0.0002
11	1000.5	3.2390	0.0379	0.0137	0.0060	0.0000
12	1192.4	2.6764	0.0350	0.0153	0.0067	0.0003
13	1401.7	9.0293	0.0243	0.0138	0.0072	0.0000
14	1628.5	7.7240	0.0170	0.0115	0.0059	0.0004
15	1873.1	15.5551	0.0039	0.0065	0.0045	0.0002
16	2135.7	13.5981	0.0038	0.0016	0.0010	0.0004
17	2416.5	19.8412	0.0178	0.0055	0.0021	0.0005
18	2715.9	17.6392	0.0246	0.0116	0.0074	0.0003
19	3033.9	21.7184	0.0404	0.0200	0.0119	0.0010
20	3370.9	19.4654	0.0451	0.0270	0.0190	0.0002
21	3727.0	24.9073	0.9385	0.0363	0.0251	0.0018
22	4102.5	22.6532	0.8858	0.0437	0.0346	0.0000
23	4497.5	42.3132	2.7518	0.0533	0.0425	0.0027
24	4912.0	38.7891	2.5073	0.5765	0.0541	0.0001
25	5346.1	76.6782	5.3295	0.5577	0.3781	0.0037

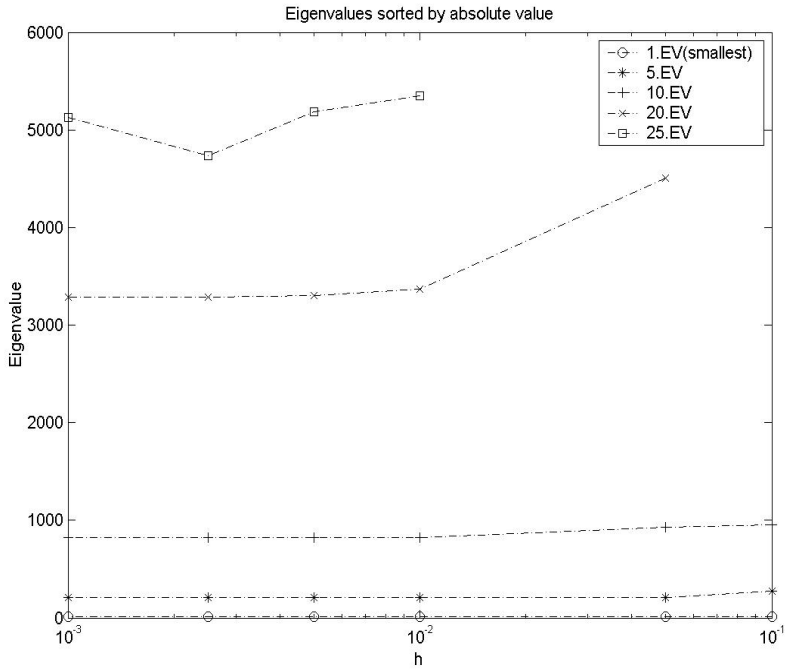

Abbildung 1: Konvergenz der reellen Eigenwerte des Neumann-EWP in Abhängigkeit der Schrittweite h.

4.2 Genauigkeit des Reduced Order Modelling (ROM) in Abhängigkeit des Grades k

Es soll die Genauigkeit der mit ROM berechneten Eigenwerte in $\lambda_{krit} \pm \delta_o \lambda_{krit}$ als Funktion des Grades k untersucht werden. In einem ersten Durchgang werden die k Basisfunktionen v_i analog dem in Kapitel 1.4 beschriebenen Algorithmus gewählt. Das heisst, die Basis für Spektral-Galerkin wird aus den Eigenmoden, deren Eigenwert am nächsten zu λ_{krit} liegt, aufgebaut. Die Eigenwerte seien aufsteigend sortiert: $\lambda_1 < \lambda_2 \ldots < \lambda_L$ mit $L = \frac{1}{h} + 1$

$$\mathcal{B}' = \{v_j, \ldots, v_{j+k-1}\} \text{ aus } (\lambda_i, v_i)_{i=1}^{L}, \text{ wobei } \sum_{i=j}^{j+k-1} |\lambda_{krit} - \lambda_i| \rightarrow min$$

Das mit ROM k-ten Grades diskretisierte EWP wird nun gelöst. Aus den resultierenden Eigenwerten müssen nun diejenigen drei y_1, y_2, y_3 ausgewählt werden, die den kritischen Bereich am besten approximieren. Das Auswahlkriterium

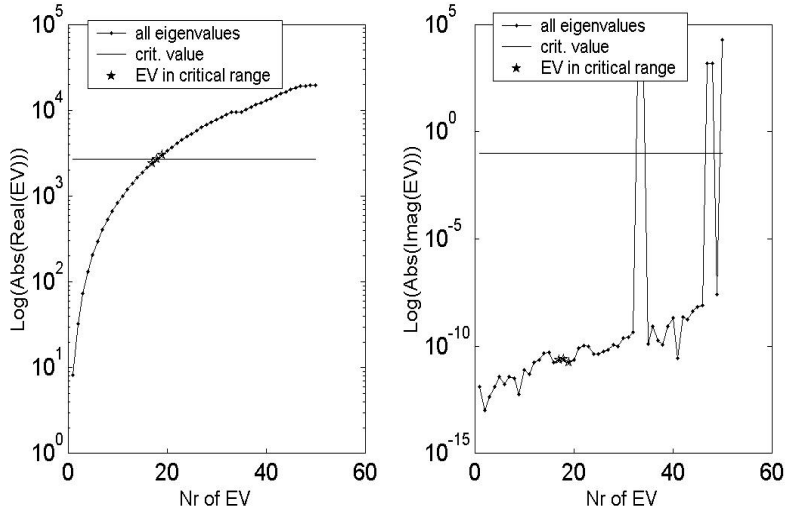

Abbildung 2: Exaktes Spektrum (Eigenwerte) für $h = 0.01$, $t = 0.1$ aufsteigend nach Realteil sortiert; zusätzlich ist der kritische Bereich mit $\lambda_{krit} = 2715$ und $\delta_0 = 0.12$ markiert

hierfür ist definiert durch

$$\sum_{i=1}^{3}(Re(z_i - y_i))^2 \rightarrow min \tag{1}$$

Das Auswahlkriterium wird hier benutzt, um festzustellen, ob die approximierten Eigenwerte überhaupt mit zunehmendem Grad k gegen die exakten Eigenwerte konvergieren. Da es die Kenntnis dieser voraussetzt, ist dieses Auswahlkriterium für praktische Anwendungen nicht brauchbar. Zur Beurteilung der Approximationsgüte ist eine a-posteriori Fehlerschätzung nötig, die einen Wert dafür liefert, wie gut die Approximationen das komplexe Eigenwertproblem lösen, und zusätzlich erlaubt, den Nutzen einer Gitterverfeinerung und/oder Graderhöhung des ROM abzuschätzen.

Abbildung (3) zeigt die ermittelten Eigenwerte in Abhängigkeit des ROM Grades k für das Modellproblem mit $h = 0.01$, $t = 0.1$, $\lambda_{krit} = 2715$ und $\delta_0 = 0.12$. Zum Vergleich wurden auch die exakten Eigenwerte eingezeichnet.

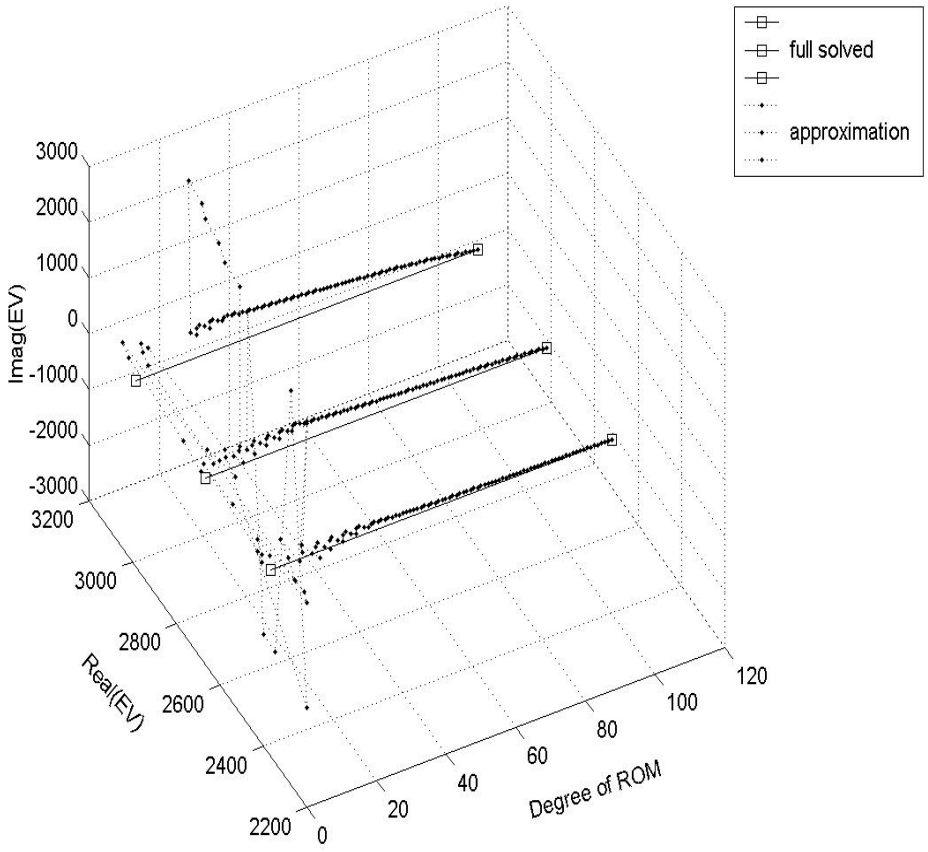

Abbildung 3: Eigenwerte mit ROM unter sukzessiver Steigerung des Grades k

Abbildung (4), (5) und (6) zeigen den Real- und Imaginärteil der approximierten Eigenwerte getrennt. Es ist deutlich zu erkennen, dass vor allem der Realteil der Eigenwerte mit zunehmendem Grad k gegen den exakten Wert konvergiert. Die Abweichung im Imaginärteil scheint willkürlich zu varieren. Es kann sein, dass approximierte Eigenwerte trotz grosser Abweichung im Imaginärteil gegenüber dem exakten Wert, unter dem Kriterium (1) ausgewählt werden, da (1) lediglich den Realteil betrachtet.

Abbildung (7) zeigt das Fehlermass r_{tot} ebenfalls in Abhängigkeit von k.
Sei der Grad k gesucht, bei welchem die Werte y_i die exakten Werte mit einer

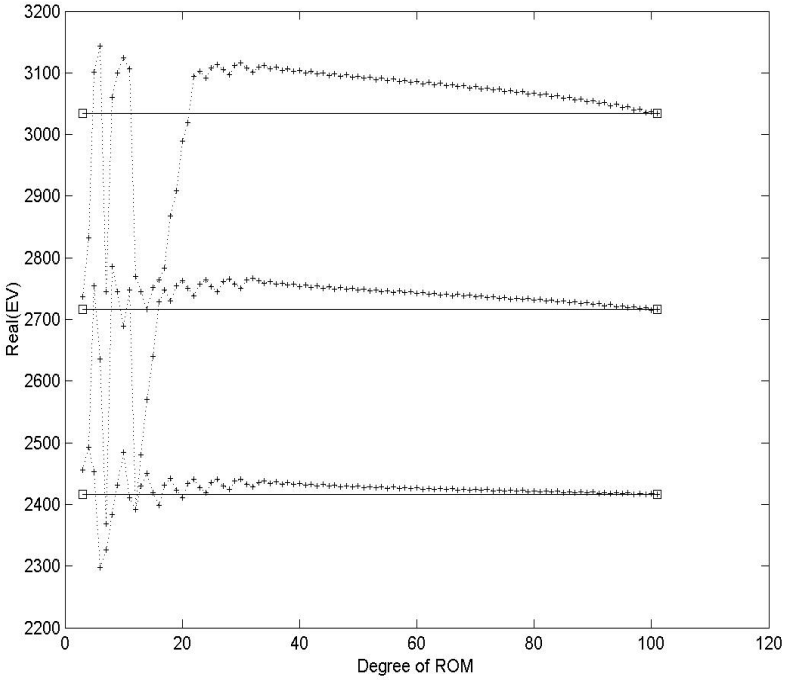

Abbildung 4: Realteil der Eigenwerte mit ROM

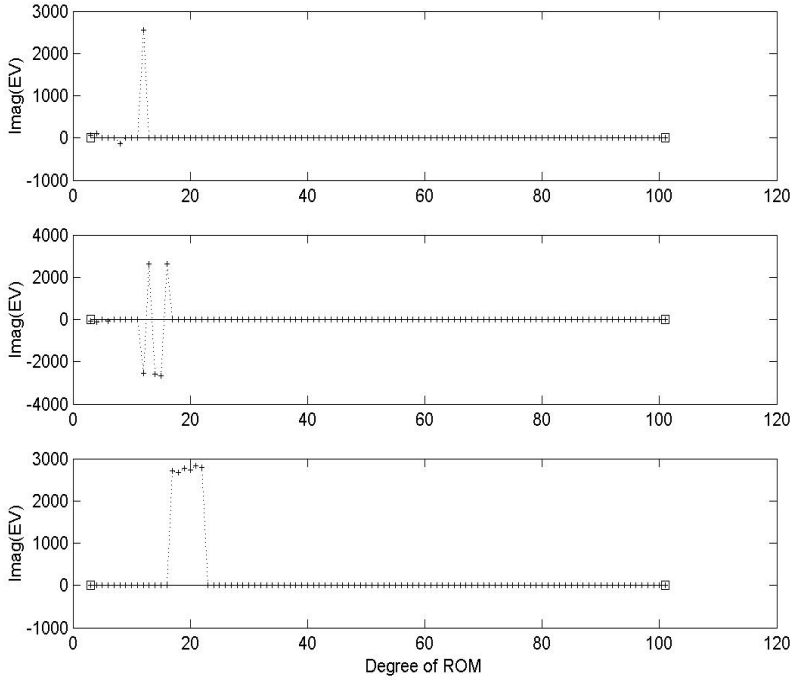

Abbildung 5: Imaginärteil der Eigenwerte mit ROM

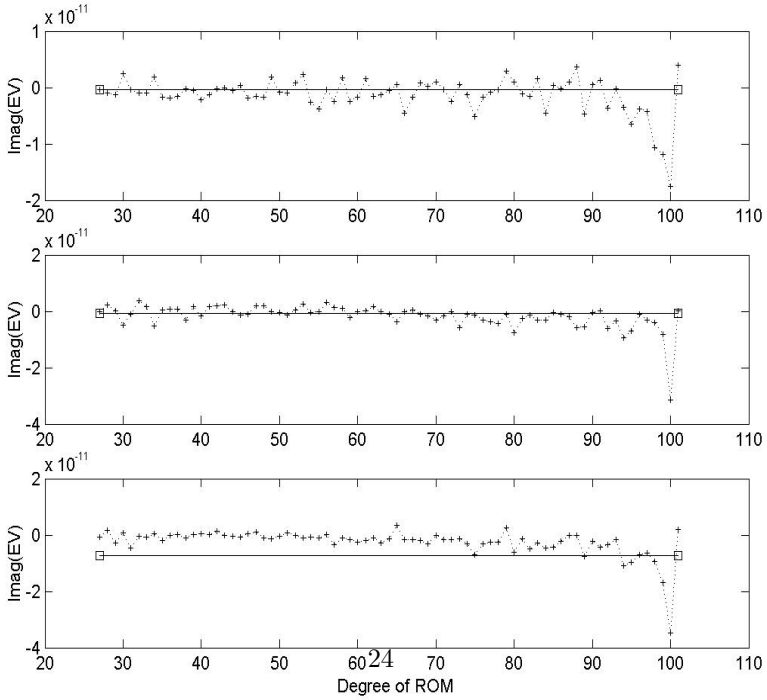

Abbildung 6: Imaginärteil der Eigenwerte mit ROM, wobei die starken Abweichungen bei kleinem k weggelassen sind

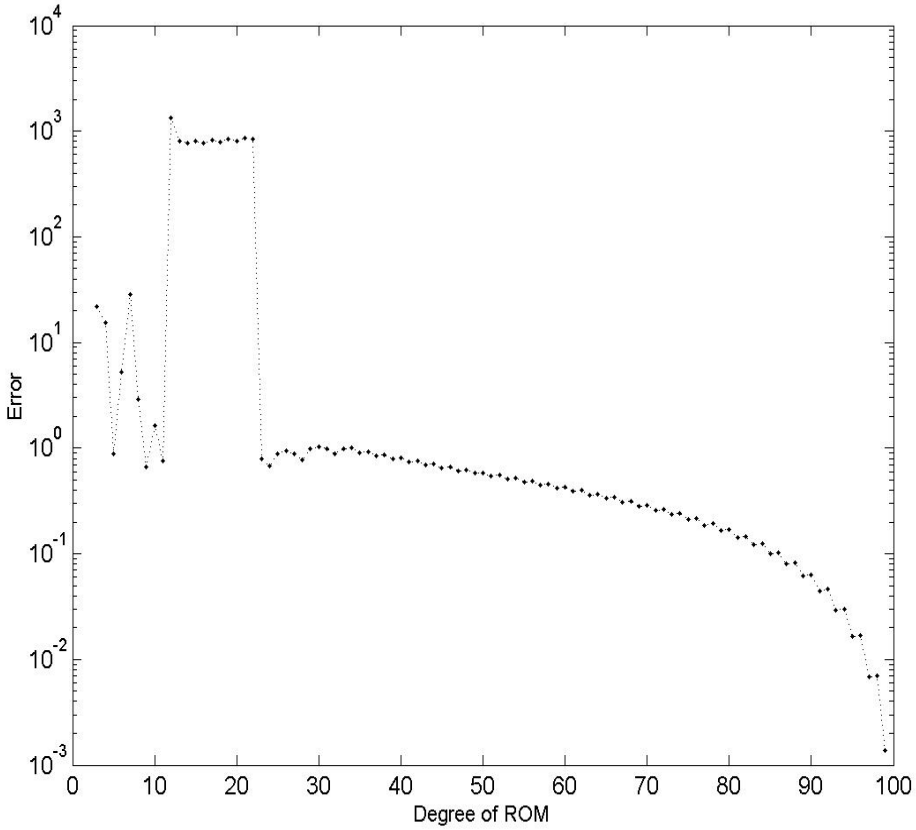

Abbildung 7: Fehler r_{tot} in Abhängigkeit von k

Genauigkeit von mindestens $\gamma = [0, 1]$ approximieren, das heisst es soll sein

$$\sum_{i=1}^{3} |z_i - y_i|^2 \leq \sum_{i=1}^{3} |z_i(1-\gamma)|^2$$

Die Resultate sind in Tabelle (2) festgehalten. Zum Vergleich wurde zusätzlich die Genauigkeit eines weiteren Testlaufes untersucht. Es ist ersichtlich, dass ein kleinerer kritischer Wert in Kombination mit einem feineren Gitter deutlich bessere Resultate liefert.

Tabelle 2: Genauigkeit γ wird ab Grad k bzw. ab % der maximalen Dimension ($k_{max} = \frac{1}{h} + 1$) erreicht.

γ	k^a[abs]	[%]	k^b[abs]	[%]
0.9	21	20.8	2	0.5
0.95	21	20.8	2	0.5
0.99	69	68.3	90	22.4
0.995	87	86.1	170	42.3
0.999	97	96.0	339	84.5
0.9995	–	–	369	92.0
0.9999	–	–	393	98.0

[a]Parameter: $h = 0.01$, $t = 0.1$, $\lambda_{krit} = 2715$, $\delta_0 = 0.12$
[b]Parameter: $h = 0.0025$, $t = 0.1$, $\lambda_{krit} = 1180$,$\delta_0 = 0.18$

4.3 Modifikationen

4.3.1 Auswahlkriterium

Das Kriterium (1) wählt die Approximationen nur nach ihrer Abweichung im Realteil aus. Da im Fehlermass aber auch der Imaginärteil wirkt, wäre ein Kriterium, das die Distanz zwischen den Eigenwerten minimiert, ein geeigneter Ansatz.

$$\sum_{i=1}^{3} \sqrt{(Re(z_i) - Re(y_i))^2 + (Im(z_i) - Im(y_i))^2} \rightarrow min \tag{2}$$

Die Abbildungen (8)-(12) zeigen das Resulat des Reduced Order Modelling mit dem Kriterium (2). Man sieht, dass das Auswahlkriterium wesentlichen Einfluss auf die Güte der Approximation hat. Mit dem Kriterium (2) werden die Abweichungen der Approximation sowohl im Real- als auch Imaginärteil deutlich abgeschwächt, bzw. es werden passendere Eigenwerte ausgewählt.

4.3.2 Wahl der Basisfunktionen

Anstelle wie in Kapitel 1.4 beschrieben, die Basisfunktionen lediglich aus Eigenmoden mit Eigenwerten nahe einem bestimmten kritischen Wert sukzessive auszuwählen, kann man schon zu Beginn Eigenmoden mit niederer Frequenz miteinbeziehen, da diese ihrerseits höhere Eigenfrequenzen besser zu approximieren vermögen.

In einem weiteren Testlauf wird diese Strategie verfolgt. Die Eigenmoden zu den kleinsten drei Eigenwerten des Neumann-EWP werden von Beginn weg als Basisfunktionen verwendet. Die übrigen Basisfunktionen (maximal $k - 3$) werden gemäss dem Algorithmus in Kapitel 1.4 eingeführt. Die Eigenwerte seien

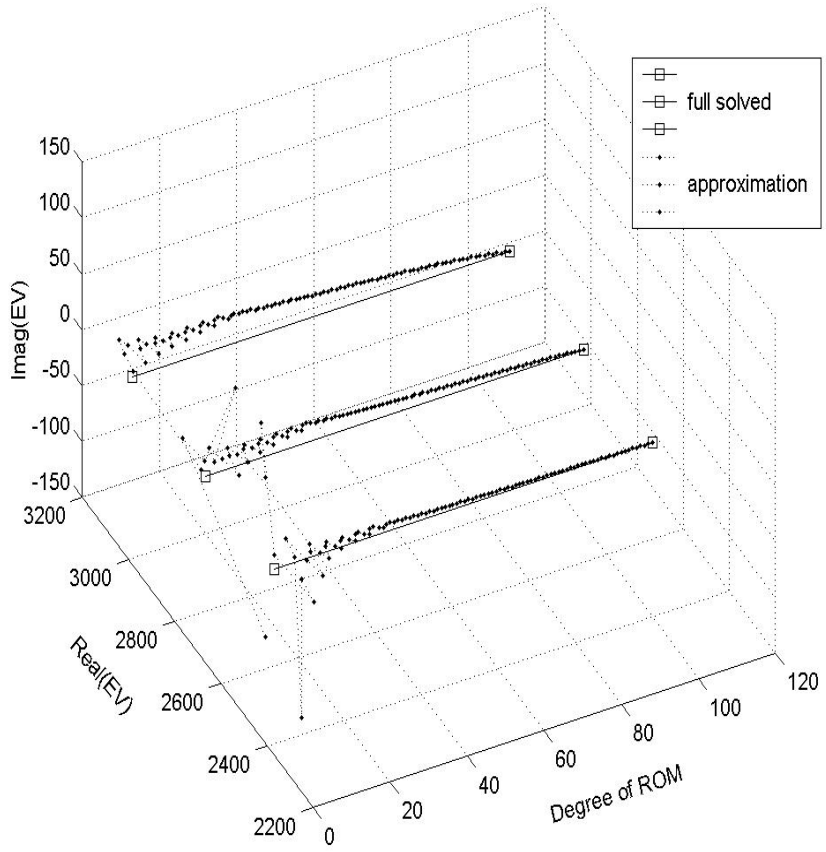

Abbildung 8: Eigenwerte mit ROM unter sukzessiver Steigerung des Grades k

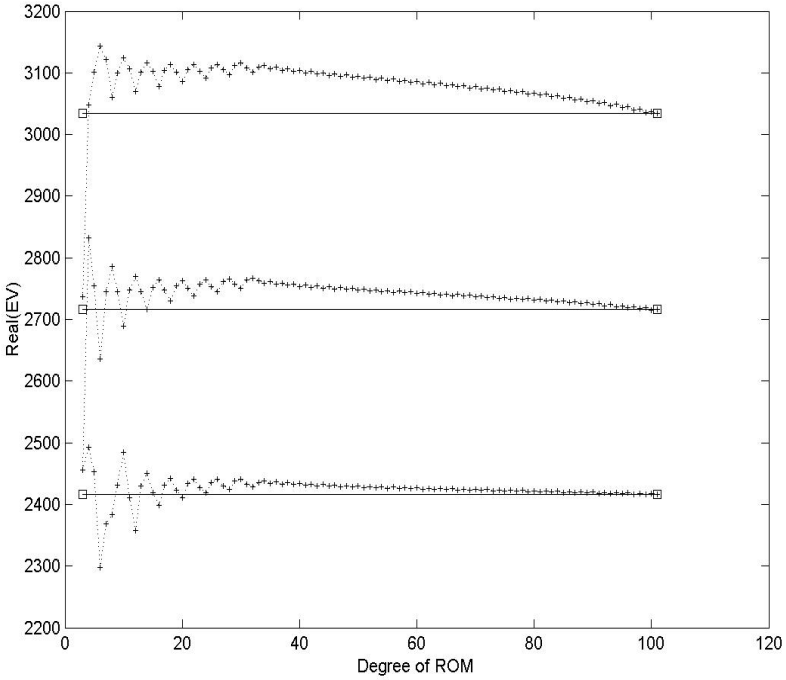

Abbildung 9: Realteil der Eigenwerte mit ROM

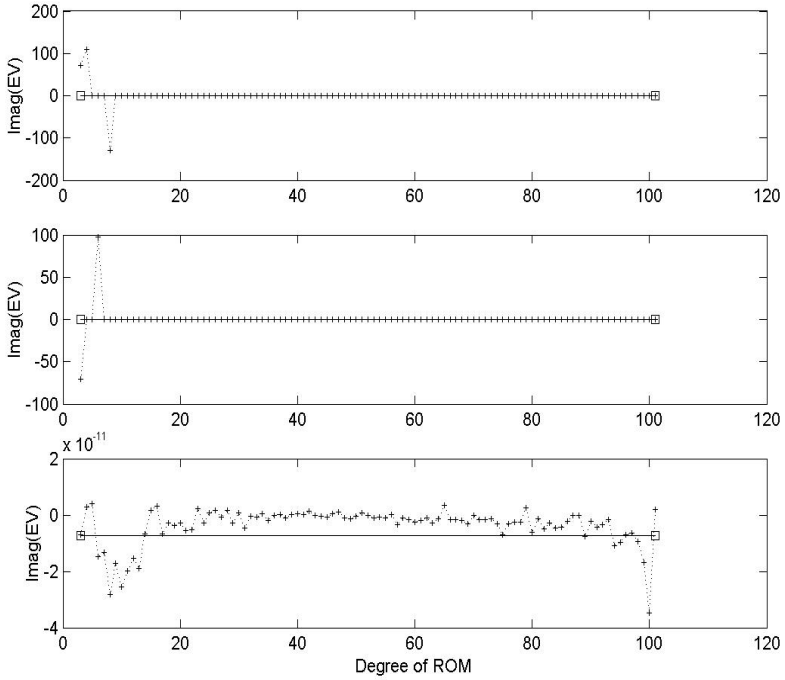

Abbildung 10: Imaginärteil der Eigenwerte mit ROM

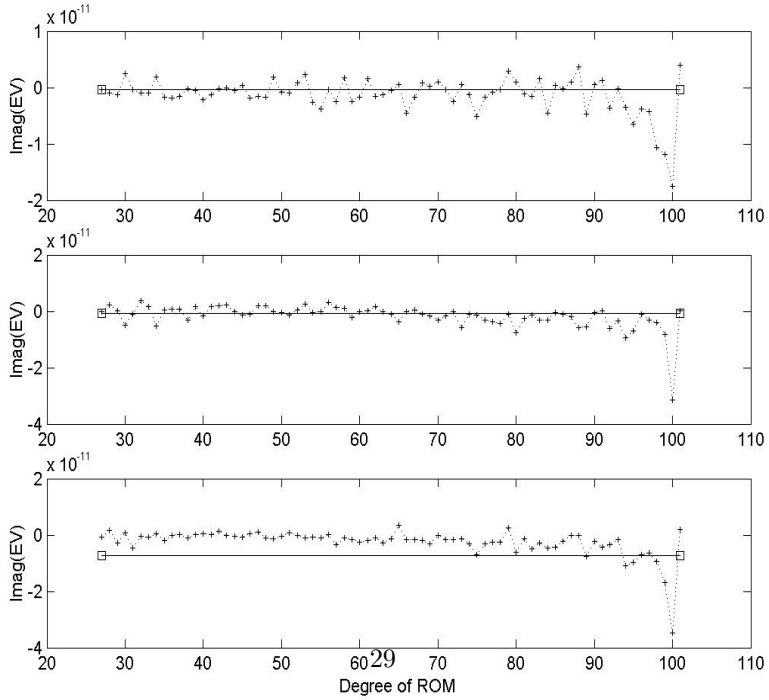

Abbildung 11: Imaginärteil der Eigenwerte mit ROM, wobei die starken Abweichungen bei kleinem k weggelassen sind

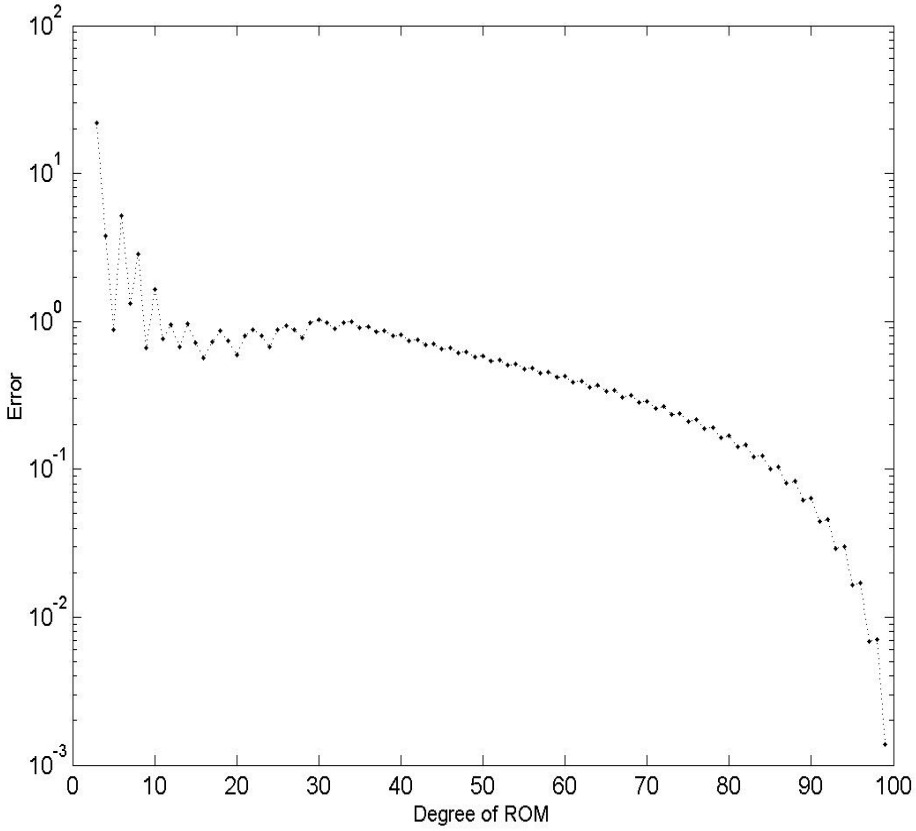

Abbildung 12: Fehler r_{tot} in Abhängigkeit von k

aufsteigend sortiert: $\lambda_1 < \lambda_2 \ldots < \lambda_L$ mit $L = \frac{1}{h} + 1$

$\mathcal{B}' = \{v_1, v_2, v_3, v_j, \ldots, v_{j+k-1}\}$ aus $(\lambda_i, v_i)_{i=4}^{L}$, wobei $\sum_{i=j}^{j+k-1} |\lambda_{krit} - \lambda_i| \rightarrow min$

Dieser Testlauf liefert keine besseren Resultate. Dies liegt unter anderem daran, dass auch bei der ursprünglichen Art der Basisbildung die niederfrequenten Eigenmoden schon bei geringem k in die Basis $\mathcal{B}'$ einfliessen, da wir λ_{krit} eher klein wählen.

5 Schlussfolgerungen und Ausblick

Zahlreiche Testläufe mit verschiedenen Schrittweiten und kritischen Bereichen haben gezeigt, dass wie erwartet vor allem kleine Eigenwerte gut approximiert werden. Für das vorgestellte Modellproblem scheint das Reduced Order Modelling k-ten Grades mit $k \sim \frac{1}{4}\frac{1}{h}$ und genügend feinem Gitter eine Approximation λ^{app} zu einem exakten klein gewählten λ_{krit}^{ex} zu liefern, für welches gilt:

$$|\lambda_{krit}^{ex} - \lambda^{app}|^2 < |0.01\lambda_{krit}^{ex}|^2$$

Die Auswahl der besten Approximationen für einen kritischen Bereich $\lambda_{krit} \pm \delta_0\lambda_{krit}$ verlangt das Testen einer Vielzahl von möglichen Kombinationen. In MATLAB müssen je nachdem, wieviele exakte Eigenwerte der Bereich von Interesse beinhaltet, verschiedene Funktionen geschrieben werden. Dies erschwert die Auswertung wesentlich bzw. schränkt die Wahl von $\lambda_{krit} \pm \delta_0\lambda_{krit}$ stark ein. Das ist der Grund, warum ich $\lambda_{krit} \pm \delta_0\lambda_{krit}$ stets so gewählt habe, dass er genau drei exakte Eigenwerte beinhaltet.

Da im Allgemeinen die exakten Eigenwerte nicht bekannt sind, wäre ein Fehlerschätzer wünschenswert. Durch die komplexe Struktur des EWP ist ein solcher aber nur schwer zu bekommen.

Um Reduced Order Modelling praktisch anwenden zu können, ist es unumgänglich auch Modellprobleme in 2d und 3d zu untersuchen.