Complex Absorbing Potential Method: theory and implementation

Samuel Edwards
## Contents

1 Introduction 2

2 The Basic Model 2
   2.1 The Riemann surface 8

3 The Complex Absorbing Potential 9

4 The Complex Symmetric Eigenvalue Problem 16
   4.1 Preliminaries 16
   4.2 The QR Algorithm 19
   4.3 The Jacobi-Davidson method 22

A Matlab code 29
   A.1 An implementation of Newton’s method to calculate the resonance energies 29
   A.2 A program that calculates the eigenvalues of $\hat{H}$ using the Complex Symmetric QR algorithm 30
   A.3 A program that calculates the trajectory of the resonance energy near $13.8 - 1.27i$ for $N = 504, L = 12$, using the Complex Symmetric Jacobi-Davidson algorithm 31
   A.4 A program that uses Householder reflections to transform a complex symmetric matrix to symmetric tridiagonal form 32
   A.5 The complex symmetric Gram-Schmidt algorithm 32
   A.6 A program that uses Givens Rotations to calculate the QR factorization of a symmetric tridiagonal matrix 33
   A.7 The complex symmetric QR algorithm 34
   A.8 The complex symmetric Jacobi-Davidson algorithm 35

B Computation of elements of $\hat{H}$ 36

Bibliography 39
1 Introduction

This report presents an example of how the complex symmetric eigenvalue problem arises when numerically calculating the resonance energies to the Schrödinger equation using the Complex Absorbing Potential method. This was done by studying the material in sections 3.1 and 4.1-4.3 of [1] and sections 1-4 of [5]. Firstly the model studied in section 3.1 of [1] is presented, the resonance energies of which are calculated semi-analytically. Then the Complex Absorbing Potential method is introduced, and a demonstration of how it can be used to find the resonance energies is given. Finally, two algorithms to compute the eigenvalues of the Complex Symmetric matrices that arise in the CAP-method are discussed. The first of these is the Complex-Symmetric QR algorithm, which is presented in sections 4.1-4.3 of [1]. The second is a Jacobi-Davidson method for Complex Symmetric matrices, as found in [5].

2 The Basic Model

The time-independent Schrödinger equation

\[ \hat{H}\psi = E\psi \]  

is considered in three dimensions, with spherical coordinates

\[ \psi = \psi(r, \theta, \phi) \]

and \( \hat{H} \) being the Hamiltonian operator

\[ \hat{H} = -\frac{1}{2} \nabla^2 + V(r). \]
The potential, $V(r)$, is defined to be spherically symmetric

$$V = \begin{cases} 
-V_0 & \text{if } 0 \leq r < a \\
V_0 & \text{if } a \leq r < 2a \\
0 & \text{if } 2a \leq r 
\end{cases}$$  \hspace{1cm} (3)$$

Expanding the Laplacian operator in spherical coordinates gives the PDE

$$-\frac{1}{2r^2} \left( \frac{\partial}{\partial r} \left( r^2 \frac{\partial}{\partial r} \right) + \frac{1}{\sin \vartheta} \frac{\partial}{\partial \vartheta} \left( \sin \vartheta \frac{\partial}{\partial \vartheta} \right) + \frac{1}{\sin^2 \vartheta} \frac{\partial^2}{\partial \varphi^2} \right) \psi + V \psi = E \psi \hspace{1cm} (4)$$

Now, to simplify the solution, the separation of variables

$$\psi = R(r)Y(\vartheta, \varphi)$$  \hspace{1cm} (5)

is used. The operator

$$\hat{L}^2 = \left( \frac{1}{\sin \vartheta} \frac{\partial}{\partial \vartheta} \left( \sin \vartheta \frac{\partial}{\partial \vartheta} \right) + \frac{1}{\sin^2 \vartheta} \frac{\partial^2}{\partial \varphi^2} \right)$$

is an angular momentum operator, the eigenfunctions of which are known as the spherical harmonics. For the sake of simplicity, so-called S-wave scattering is studied, i.e. the special case $\hat{L}^2 Y(\vartheta, \varphi) = 0$. In this case $Y$ will be a non-zero constant which, in combination with $\hat{L}^2 R(r) = 0$, gives

$$-\frac{1}{2r^2} \frac{d}{dr} \left( r^2 \frac{d}{dr} \right) R(r) + V(r)R(r) = E R(r) \hspace{1cm} (6)$$

This ordinary differential equation is solved with the help of the substitution $R(r) = \frac{u(r)}{r^2}$. This yields

$$-\frac{1}{2r} \left( 2 \frac{d}{dr} + r^2 \frac{d^2}{dr^2} \right) \frac{u(r)}{r} + V(r)u(r) = Eu(r)$$

$$-\frac{1}{2} \left( 2 \left( -\frac{u}{r^2} + \frac{u'}{r} \right) + \left( -\frac{u'}{r^2} + 2 \frac{u}{r^3} + \frac{u''}{r} - \frac{u'}{r^2} \right) \right) + Vu = Eu$$

Which simplifies to

$$u'' + 2(E - V)u = 0 \hspace{1cm} (7)$$

This differential equation is then solved in the regions where $V(r)$ assumes different values. Continuity of $u$ and $u'$ at the boundary between regions
is required. Furthermore, to avoid singularities in $R(r)$, it is required that $u(0) = 0$, as

$$\lim_{r \to 0} R(r) = \lim_{r \to 0} \frac{u(r)}{r}$$

The solution of the equation is

$$u(r) = A_1(e^{i\sqrt{2(E+V_0)r}} - e^{-i\sqrt{2(E+V_0)r}}) \quad 0 \leq r < a$$

$$u(r) = A_2e^{i\sqrt{2(E-V_0)r}} + A_3e^{-i\sqrt{2(E-V_0)r}} \quad a \leq r < 2a$$

$$u(r) = A_4e^{i\sqrt{2E}r} + A_5e^{-i\sqrt{2E}r} \quad 2a \leq r$$

The continuity requirements at $r = a$ give the following relations

$$2A_1i \sin(k_1a) = A_2e^{ik_2a} + A_3e^{-ik_2a} \quad (9)$$

$$2A_1ik_1 \cos(k_1a) = A_2ik_2e^{ik_2a} - A_3ik_2e^{-ik_2a} \quad (10)$$

$$(k_1 = \sqrt{2(E+V_0)}, \quad k_2 = \sqrt{2(E-V_0)}, \quad k_3 = \sqrt{2E})$$

Solving this system for $A_2$ and $A_3$ gives

$$A_2 = A_1e^{-ik_2a}(i \sin(k_1a) + \frac{k_1}{k_2} \cos(k_1a)) \quad (11)$$

$$A_3 = A_1e^{ik_2a}(i \sin(k_1a) - \frac{k_1}{k_2} \cos(k_1a)) \quad (12)$$

Likewise, at $r = 2a$, the system to be solved is

$$A_2e^{2ik_2a} + A_3e^{-2ik_2a} = A_4e^{2ik_3a} + A_5e^{-2ik_3a} \quad (13)$$

$$A_2ik_2e^{2ik_2a} - A_3ik_2e^{-2ik_2a} = A_4ik_3e^{2ik_3a} - A_5ik_3e^{-2ik_3a} \quad (14)$$

Solving for $A_4$ and $A_5$ in terms of $A_2$ and $A_3$ gives

$$A_4 = e^{-2ik_3a} \frac{1}{2} \left( A_2e^{2ik_2a} \left( 1 + \frac{k_2}{k_3} \right) + A_3e^{-2ik_2a} \left( 1 - \frac{k_2}{k_3} \right) \right) \quad (15)$$

$$A_5 = e^{2ik_3a} \frac{1}{2} \left( A_2e^{2ik_2a} \left( 1 - \frac{k_2}{k_3} \right) + A_3e^{-2ik_2a} \left( 1 + \frac{k_2}{k_3} \right) \right) \quad (16)$$

The energies that are desired are the Siegert resonance energies (see [2]). These are the energies that correspond to poles of the so-called S-matrix, $S(E)$, in the fourth quadrant of the complex plane

$$S(E) := \frac{A_4}{A_5} \quad (17)$$
with the wave function being an outgoing wave. This can be formulated equivalently as the energies were $A_5 = 0$ and the wave function for $r \geq 2a$ is

$$u(r) = A_4 e^{ik_3}, \Re(k_3) \geq 0$$

To locate these energies, $A_5$ is expressed as a function of $E$ by inserting (11) and (12) into (16):

$$A_5 = e^{2ik_3a} \frac{1}{2} \left( A_2 e^{2ik_2a} \left( 1 - \frac{k_2}{k_3} \right) + A_3 e^{-2ik_2a} \left( 1 + \frac{k_2}{k_3} \right) \right)$$

$$A_5 = \frac{e^{ik_2a}}{2} \left( i \sin(k_1a) + \frac{k_1}{k_2} \cos(k_1a) \right) \left( 1 - \frac{k_2}{k_3} \right) + \frac{e^{-ik_2a}}{2} \left( i \sin(k_1a) - \frac{k_1}{k_2} \cos(k_1a) \right) \left( 1 + \frac{k_2}{k_3} \right) = 0$$

Euler’s formula then gives

$$A_5 = \frac{e^{ia(k_1+k_2)}}{4} \left( 1 + \frac{k_1}{k_2} - \frac{k_2}{k_3} - \frac{k_1}{k_3} \right) + \frac{e^{ia(k_2-k_1)}}{4} \left( -1 + \frac{k_1}{k_2} + \frac{k_2}{k_3} - \frac{k_1}{k_3} \right) + \frac{e^{-ia(k_1+k_2)}}{4} \left( -1 - \frac{k_1}{k_2} - \frac{k_2}{k_3} - \frac{k_1}{k_3} \right) + \frac{e^{-ia(k_2-k_1)}}{4} \left( 1 - \frac{k_1}{k_2} + \frac{k_2}{k_3} - \frac{k_1}{k_3} \right) = 0 \quad (18)$$

and again, with the addition formulae for sinus and cosinus

$$i \sin k_1a \cos k_2a + \frac{i k_1}{k_2} \sin k_2a \cos k_1a + \frac{k_2}{k_3} \sin k_1a \sin k_2a - \frac{k_1}{k_3} \cos k_1a \cos k_2a = 0$$

Which simplifies to

$$k_2k_3 \sin k_1a \cos k_2a + k_1k_3 \sin k_2a \cos k_1a + ik_1k_2 \cos k_1a \cos k_2a - i k_2^2 \sin k_1 \sin k_2 = 0 \quad (19)$$

After expanding $k_1, k_2, k_3$ in terms of $E$ and $V_0$, and letting the sum of the terms define the function $P(E)$, the problem of finding the relevant poles of
$S(E)$ is reduced to finding the zeroes of $P(E)$.

\[ P(E) := \sqrt{(E^2 - EV_0)} \sin(\sqrt{2(E + V_0)a})\cos(\sqrt{2(E - V_0)a}) + \]
\[ \sqrt{(E^2 + EV_0)} \sin(\sqrt{2(E - V_0)a})\cos(\sqrt{2(E + V_0)a}) + \]
\[ i\sqrt{(E^2 - V_0^2)} \cos(\sqrt{2(E + V_0)a})\cos(\sqrt{2(E - V_0)a}) - \]
\[ i(E - V_0) \sin(\sqrt{2(E + V_0)a})\sin(\sqrt{2(E - V_0)a}) \]

Locating the zeroes of $P(E)$ is somewhat difficult to solve analytically, so, as in [1], the zeros are computed numerically, using Newton’s method

\[ E_{j+1} = E_j - \frac{P(E_j)}{P'(E_j)} \]

in the specific case $a = 1, V_0 = 10$. The results are collected in Table 1. See Appendix A for the Matlab program used for this. The energy at $-6.3538$ is a bound state, that is

\[ \int_0^\infty |u(r)|^2 dr < \infty \]

as, for energies in $\mathbb{R}_-$, the function $A_4e^{ik_3}$ is exponentially decaying. Also of note is the imaginary part of $E$, known as the width of the resonance. The width describes the decay of the resonance state with respect to time, as seen by studying the time dependent Schrödinger equation

\[ i\frac{\partial}{\partial t} \psi = \hat{H}\psi \]

This is solved using the solution from the time dependant equation, $R(r)$, together with an exponential term

\[ \psi(r,t) = R(r)e^{-iEt} \]

In the case $E$ is a resonance energy, $E = E_R - i\Gamma/2 \ (E_R, \Gamma \in \mathbb{R}_+)$, giving

\[ \psi(r,t) = R(r)e^{-i(E_R-i\Gamma/2)t} = R(r)e^{-iE_Rt-\Gamma/2} \]

Using the values from Table 1, we see that the width of the resonance increases as $E$ increases. This is explained thus, when $\Re(E) < V_0$, the potential barrier traps the particle (described by the wave function) more effectively than for the higher energy levels, giving a more stable state. From Figure 1, it would appear that the resonance energies could have radial density functions, $|u(r)|^2$, that might be in $L^2$. This, however, is not the case, as is demonstrated in the next section.
$-6.353800491353072 + 0.000000476966992i$
$4.001414397251380 - 0.003616371422452i$
$13.804342496156762 - 1.269152015401677i$
$20.677306105302716 - 2.065452505760665i$
$30.560595076771026 - 5.020902968718982i$
$45.230930043696411 - 6.030018368838527i$
$59.163906855958253 - 8.627586505485850i$
$79.476677197973032 - 10.610235576861449i$
$98.273816336887705 - 12.612523040649435i$

Table 1: Calculated values of $E$, $|\Re(E)| < 100$. The initial values were integers between $-100$ and $100$, with the exceptions of $-10$, $0$, and $10$.

Figure 1: $|u(r)|^2$ for various values of $E$
2.1 The Riemann surface

A study of \( u(r) \) for large values of \( r \), \( r \geq 2a \), where \( E \) is a resonance energy, \( E = E_R - i\Gamma/2 \) \( (E_R, \Gamma \in \mathbb{R}_+) \), leads to a discussion of the nature of the root function in the complex plane. In the situation described above, the wave function has the form

\[
u(r) = A_4 e^{i \sqrt{2(E_R - i\Gamma/2)} r}
\] (20)

The square root is multivalued, and therefore its definition is somewhat ambiguous. Using the principal branch

\[
\sqrt{E} := \sqrt{|E|} e^{i\theta/2} \quad E = |E| e^{i\theta}, \theta \in [0, 2\pi)
\]
gives rise to standard problems, continuity at the positive real numbers, for example;

\[
\lim_{\theta \to 0} \sqrt{|E|} e^{i\theta/2} = \sqrt{|E|} \neq \lim_{\theta \to 2\pi} \sqrt{|E|} e^{i\theta/2} = -\sqrt{|E|}
\]

The issues that arise when dealing with multivalued functions are resolved by introducing the notion of a Riemann surface, namely by allowing values of \( \theta \) outside the interval \([0, 2\pi)\). In this way, \( \sqrt{E} := \sqrt{|E|} e^{i\theta/2} \) now defines a continuous, single-valued function on the Riemann surface consisting of 2 sheets, which are known as the “physical” sheet; \( \theta \in [4\pi n, 2\pi + 4\pi n), n \in \mathbb{Z} \) and the “non-physical” sheet; \( \theta \in [-2\pi + 4\pi n, 4\pi n), n \in \mathbb{Z} \). Returning to the wave function in (20), this being an outgoing wave gives the requirement that \( \Re(\sqrt{2E}) > 0 \), allowing the determination of which of the two sheets of the Riemann surface \( E \) resides on. If \( E \) lies on the physical sheet, then

\[
\theta = 2\pi - \arctan(\Gamma/(2E_R))
\]

Which gives

\[
\Re(e^{i\frac{1}{2}(2\pi - \arctan(\Gamma/(2E_R)))}) = \cos(\pi - \frac{1}{2} \arctan(\Gamma/(2E_R))) < \cos(\pi - \frac{1}{2} \times \frac{\pi}{2}) < 0
\]

Hence, \( E \) must lie on the non-physical sheet, and \( \theta = - \arctan(\Gamma/(2E_R)) \). \( k_3 \) is then calculated

\[
k_3 = \sqrt{2E} = \sqrt{2} \sqrt{E_R^2 + \Gamma^2/4} \left( \cos \left( \frac{1}{2} \arctan(\Gamma/(2E_R)) \right) - i \sin \left( \frac{1}{2} \arctan(\Gamma/(2E_R)) \right) \right)
\]
It is now noted that $\Re(k_3)$ is positive and $\Im(k_3)$ is negative. Returning to $u(r), \quad r \geq 2a$

\[ u(r) = A_4 e^{i r (\Re(k_3) - \Im(k_3))} = A_4 e^{r |\Im(k_3)| + i r \Re(k_3)} \]

The radial density function, $|u(r)|^2$, is then

\[ |u(r)|^2 = A_4 e^{r |\Im(k_3)| + i r \Re(k_3)} A_4 e^{r |\Im(k_3)| - i r \Re(k_3)} = A_4^2 e^{2r |\Im(k_3)|} \]

As $|\Im(k_3)|$ is positive, this is exponentially divergent, and hence $u$ definitely does not belong to $L^2$.

3 The Complex Absorbing Potential

In an attempt to make the radial density functions $L^2$, the Hamiltonian, $\hat{H}$, is now modified somewhat. The new operator is defined by

\[ \hat{H}(\eta) := \hat{H} - i\eta \hat{W} \] (21)

The term $-i\eta \hat{W}$ is the Complex Absorbing Potential, (CAP). A demonstration of how the CAP can be used to calculate the resonance energies is now given. We start with a complex vector space of continuous functions whose support is a subset of the interval $(0, L)$. The standard inner product on such a space is used, namely

\[ \langle u, v \rangle = \int_0^\infty u(r)v(r)dr \]

To numerically compute the eigenvalues of the operator $\hat{H}(\eta)$, a finite orthonormal basis, $\{\phi_n\}_{n=1,2,3...N}$ is used. In this finite dimensional vector space, $\hat{H}(\eta)$ is represented by a matrix, $\hat{H}(\eta)$. The orthonormality of the basis gives the elements of $\hat{H}(\eta)$;

\[ \hat{H}(\eta)_{j,k} = \langle \hat{H}(\eta)\phi_k, \phi_j \rangle \] (22)

The elements of the chosen basis are defined thus

\[ \phi_n(r) = \begin{cases} \sqrt{2/L} \sin(n\pi r/L) & \text{if } 0 \leq r < L \\ 0 & \text{if } 2a \leq r \end{cases} \] (23)
Orthonormality follows from various trigonometric identities

\[ \langle \phi_j, \phi_j \rangle = \frac{2}{L} \int_0^L \sin^2(j\pi r/L) dr = \frac{1}{L} \int_0^L 1 - \cos(2j\pi r/L) dr = 1 - \frac{1}{2\pi j} (\sin(2\pi j) - \sin(0)) = 1 \]

\[ \langle \phi_j, \phi_k \rangle = \frac{2}{L} \int_0^L \sin(j\pi r/L) \sin(k\pi r/L) dr = \frac{1}{L} \int_0^L \cos((j - k)\pi r/L) - \cos((j + k)\pi r/L) dr \]

\[ = \left[ \frac{1}{\pi(j - k)} \sin((j - k)\pi r/L) - \frac{1}{(j + k)\pi} \sin((j + k)\pi r/L) \right]_r=0 = 0 \]

The CAP to be used in this case is defined

\[ \hat{W}(r) = \begin{cases} 
0 & \text{if } 0 \leq r < 2a \\
(r - 2a)^2 & \text{if } 2a \leq r 
\end{cases} \]  \( (24) \)

The elements of \( \hat{H}(\eta) \) are now calculated. (See Appendix B for full details of these calculations)

\[ \hat{H}(\eta)_{j,k} = \int_0^\infty \phi_j(r) \left( -\frac{1}{2} \frac{d^2}{dr^2} + V(r) - i\eta \hat{W}(r) \right) \phi_k(r) dr \]  \( (25) \)

The values for the non-diagonal elements are

\[ \hat{H}(\eta)_{j,k} = \frac{2V_0}{\pi(k+j)} \sin \left( \frac{\pi(k+j)a}{L} \right) - \frac{2V_0}{\pi(k-j)} \sin \left( \frac{\pi(k-j)a}{L} \right) + \]

\[ \frac{V_0}{\pi(k-j)} \sin \left( \frac{2\pi(k-j)a}{L} \right) - \frac{V_0}{\pi(k+j)} \sin \left( \frac{2\pi(k+j)a}{L} \right) + \]

\[ -i\eta \left( \frac{2L(L-2a)}{\pi^2(k-j)^2} (-1)^{k-j} - \frac{2L(L-2a)}{\pi^2(k+j)^2} (-1)^{k+j} \right) + \]

\[ -i\eta \left( \frac{2L^2}{\pi^3(k-j)^3} \sin \left( \frac{2\pi(k-j)a}{L} \right) - \frac{2L^2}{\pi^3(k+j)^3} \sin \left( \frac{2\pi(k+j)a}{L} \right) \right) \]

And the diagonal elements

\[ \hat{H}(\eta)_{k,k} = \frac{\pi^2k^2}{2L^2} + \frac{V_0}{\pi k} \sin \left( \frac{2\pi ka}{L} \right) - \frac{V_0}{2\pi k} \sin \left( \frac{4\pi ka}{L} \right) + \]

\[ -i\eta \left( \frac{(L-2a)^3}{3L} - \frac{L(L-2a)}{2\pi^2k^2} - \frac{L^2}{4\pi^3k^3} \sin \left( \frac{4\pi ka}{L} \right) \right) \]
Figure 2: Eigenvalues of $\hat{H}(\eta)$ for a few values of $\eta$ with $N = 800, L = 16$

From these calculations, it is seen that $\hat{H}(\eta)$ is symmetric, i.e. $\hat{H}(\eta)_{j,k} = \hat{H}(\eta)_{k,j}$, or to be more precise, a complex symmetric matrix. Section 4 presents some algorithms and the theoretical aspects for calculating the eigenvalues. The spectrum of $\hat{H}(\eta)$ is computed numerically. Fig. 2 shows the eigenvalues for four different values of $\eta$ near 0. It appears that as $\eta$ increases, the eigenvalues that do not correspond to resonance states get shifted, and allows a determination of the resonance energies, which appear to stabilize after $\eta$ reaches a certain level. The effect of the CAP on the radial density function is clearly seen in Fig. 3; the distortion of the function appears to be small outside the region where the CAP acts, and the CAP “tames” the function where it is active. Table 2 demonstrates that, as might be expected, the eigenvalues of $\hat{H}(\eta)$ are not exactly the resonance energies calculated in Table 1. This leads to the question of how changes in $\eta$ effect the eigenvalue of $\hat{H}(\eta)$ that corresponds to a resonance energy.
Figure 3: $|u(r)|^2$ for $E = 13.8 - 1.27i$ with the CAP “off” and “on”

Resonance eigenvalue of $\hat{H}(\eta)$

<table>
<thead>
<tr>
<th>$E_{\hat{H}} - E_{Res}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.001304829110570 − 0.002992360081990i</td>
</tr>
<tr>
<td>13.803438894696464 − 1.265716781056169i</td>
</tr>
<tr>
<td>20.678765157594302 − 2.064682172871625i</td>
</tr>
</tbody>
</table>

Table 2: Calculated Eigenvalues of $\hat{H}(\eta)$, with $N = 800, L = 16, \eta = 0.089$. These are then compared with the resonance energies $E_{Res}$ calculated in Table 1.
The eigenvalues for the third resonance energy (roughly $13.8 - 1.27i$), with $N = 504, L = 12$ are computed for values of $\eta$ that are increased according to the formula

$$\eta = 0.005 \times \left(1.1^{\frac{n-1}{3}}\right) \quad n = 1, 2, 3, \ldots, 301$$

The trajectory is plotted in Fig. 4. It appears that the eigenvalue approaches, and stabilizes near, the resonance and then moves away again. Considering $E$ as a differentiable function of $\eta$, the resonance energy can be considered as $E(0)$. A series expansion of $E(\eta)$ can be used to find the value of $\eta$ that minimizes the distortion to the resonance by the CAP

$$E(0) = E(\eta) - \eta E'(\eta) + O(\eta^2)$$

$|\eta E'(\eta)|$ achieves its minimum near 0.12374, (see Fig. 6), with the values

$$E(0.12374) \approx 13.80341 - 1.26484i$$
$$0.12374 \times E'(0.12374) \approx -0.00203 + 0.00356i$$
Applying the first-order correction gives

\[ E(0) \approx E(0.12374) - 0.12374 \times E'(0.12374) \]

\[ \approx 13.805433610829905 - 1.26840442123181i \]

The absolute error in this value, compared with the value in Table 1, is 1.32 × 10⁻³, which is better than the value achieved in Table 2, where a greater basis set is used. This minimization and first-order correction process is completed for a few other resonance energies and collected in Table 3.

This concludes the discussion of the Complex Absorbing Potential method, the full mathematical justification of which is beyond the scope of this report, and is therefore omitted. We now proceed with a presentation and discussion of algorithms related to the calculation of eigenvalues for a complex symmetric matrix.
First-order corrected eigenvalue of $\hat{H}(\eta)$ & $\eta$ & Absolute error \\
4.001535183266297 − 0.003593251541476$i$ & 0.027796 & $1.22 \times 10^{-4}$ \\
13.805433610829905 − 1.268404442123181$i$ & 0.12374 & $1.32 \times 10^{-3}$ \\
20.678182659325380 − 2.065950520961411$i$ & 0.18117 & $1.00 \times 10^{-3}$ \\

Table 3: First-order corrected Eigenvalues of $\hat{H}(\eta)$, with $N = 504, L = 12$, and the corresponding optimized value for $\eta$. 

Figure 6: $|\eta E'(\eta)|$
4 The Complex Symmetric Eigenvalue Problem

The matrices that arise in the CAP computations are complex symmetric, i.e.

\[ A \in \mathbb{C}^{n \times n}, \quad A = A^T \]

Unfortunately, in general, complex symmetric matrices are not Hermitian, and are not necessarily non-defective. In the following section it is generally assumed that the CS-matrices that are being studied are fully diagonalizable. Since calculating the eigenvalues of a matrix by solving the characteristic equation is numerically unstable and inaccurate for large matrices, other methods are required. Two algorithms are presented; the Complex Symmetric QR Algorithm, from section 4 of [1], for calculating the full spectrum and eigenvectors of a CS-matrix, and the Complex Symmetric Jacobi-Davidson Algorithm, for approximating the eigenvalue closest to a given value (as shown in [5]).

4.1 Preliminaries

In light of the the fact that

\[(Ax)^T = x^T A^T = x^T A\]

a change to the standard inner product on \( \mathbb{C}^n \), from

\[ \langle x, y \rangle = y^\dagger x \]

to the symmetric form

\[ \langle x, y \rangle_T := y^T x \]

permits the use of the structure of complex symmetric matrices in the following way;

\[ \langle Ax, y \rangle_T = y^T A x = (y^T A) x = (Ay)^T x = \langle x, Ay \rangle_T \]

This, however, occurs at a cost, namely the loss of the norm;

\[ \langle x, x \rangle_T = 0 \quad \Rightarrow \quad x = 0 \]
and $\langle \mathbf{x}, \mathbf{x} \rangle_T$ need not even be real. Despite these deficiencies, the following definitions can still be used; Two vectors, $\mathbf{x}$, and $\mathbf{y}$ are said to be orthogonal if

$$\langle \mathbf{x}, \mathbf{y} \rangle_T = 0$$

and a set of vectors $\{ \mathbf{e}_i \}$ is said to be orthonormal if

$$\langle \mathbf{e}_j, \mathbf{e}_k \rangle_T = \delta_{jk}$$

A simple theorem can now be stated

**Theorem 1.** The eigenvectors of a complex symmetric matrix that correspond to different eigenvalues are orthogonal.

**Proof.** Let $A\mathbf{x} = \lambda_1 \mathbf{x}$ and $A\mathbf{y} = \lambda_2 \mathbf{y}$, with $\lambda_1 \neq \lambda_2$. Then

$$\lambda_1 \langle \mathbf{x}, \mathbf{y} \rangle_T = \langle A\mathbf{x}, \mathbf{y} \rangle_T = \langle \mathbf{x}, A\mathbf{y} \rangle_T = \lambda_2 \langle \mathbf{x}, \mathbf{y} \rangle_T$$

$$\Rightarrow (\lambda_1 - \lambda_2) \langle \mathbf{x}, \mathbf{y} \rangle_T = 0 \Rightarrow \langle \mathbf{x}, \mathbf{y} \rangle_T = 0$$

Another definition is useful; a matrix $Q \in \mathbb{C}^{n \times n}$ is called complex orthogonal if $Q^TQ = I$

The column vectors, $\{ \mathbf{q}_i \}$, of $Q$ form an orthonormal basis of $\mathbb{C}^{n \times n}$, as

$$\langle \mathbf{q}_k, \mathbf{q}_j \rangle_T = \mathbf{q}_j^T \mathbf{q}_k = (Q^TQ)_{j,k} = \delta_{j,k}$$

A similarity transformation of a complex symmetric matrix, $A$, by a complex orthogonal matrix $Q$, is defined as

$$Q^T AQ$$

Similarity transformations by complex orthogonal matrices preserve eigenvalues as

$$\det(Q^T AQ - \lambda I) = \det(Q^T AQ - Q^T \lambda IQ) = \det(Q^T(A - \lambda I)Q) = \det(A - \lambda I)$$

and preserve the complex symmetry

$$\left( Q^T AQ \right)^T = Q^T A^T (Q^T)^T = Q^T AQ$$

In the QR algorithm, two special types of orthogonal transformations are used, Householder reflections and Givens rotations.
Householder Reflections

The Householder reflection is defined as

\[ Hx = (I - 2vv^T)x \]

for a given unit vector, \( v \), \((v, v)_T = 1\). The transformation matrix is complex symmetric and orthogonal as

\[ (I - 2vv^T)^T = I - (2vv^T)^T = I - 2vv^T \]

and

\[ (I - 2vv^T)^2 = I - 4vv^T + 4v(v^Tv)v^T = I \]

The implementation of the QR algorithm in section 4.2 uses Householder reflections in the following way; given a vector \( x \), find \( v \) so that

\[ Hx = ke_1 \]

where \( e_1 \) is the first basis vector in the standard basis \((e_1 = (1, 0, 0, 0, ...)^T)\).

This gives

\[ k^2 = (Hx)^THx = x^TH^THx = x^Tx \]

Which allows a choice of \( k \), \((k = \pm\sqrt{x^Tx})\), provided \( x^Tx \neq 0 \). Denoting the first element of \( x \) as \( x_1 = x^Te_1 \), a solution for \( v \) can be found after noting that

\[
(x-ke_1)(x-ke_1)^Tx = (xx^T-kxe_1^T-ke_1x^T+k^2e_1e_1^T)x = (k^2-kx_1)x+(k^2x_1-k^3)e_1
\]

This means that

\[
\left( I - 2\frac{(x-ke_1)(x-ke_1)^T}{2(k^2-kx_1)} \right)x = ke_1
\]

and \( v \) is found

\[
v = \frac{x-ke_1}{\sqrt{2(k^2-kx_1)}}
\]

(27)

To make sure that this expression is always defined, if \( x_1^2 = x^Tx = k^2 \), then the sign of \( k \) is chosen so that \( k = -x_1 \).
Givens Rotations

Another type of complex orthogonal matrices are the Givens Rotations, which have the structure of the identity map, apart from four elements.

\[
G(s, c, j, k) = \begin{pmatrix}
1 & 0 & \cdots & \cdots & \cdots & 0 \\
0 & 1 & \cdots & \cdots & \cdots & \\
\cdots & \cdots & c & s & \cdots & \\
\cdots & \cdots & \cdots & \cdots & \cdots & \\
\cdots & \cdots & \cdots & \cdots & \cdots & \\
0 & \cdots & \cdots & \cdots & \cdots & 1
\end{pmatrix}
\]

\[
G(s, c, j, k)_{m,n} =
\begin{cases}
  c, & \text{if } (m, n) = (j, j) \\
  c, & \text{if } (m, n) = (k, k) \\
  s, & \text{if } (m, n) = (j, k) \\
  -s, & \text{if } (m, n) = (k, j) \\
  \delta_{mn}, & \text{otherwise}
\end{cases}
\]

To make the matrix orthogonal it is required that \(c^2 + s^2 = 1\). Calculating the product \(GA\) is fairly straightforward, all elements of \(GA\) are equal to the corresponding elements of \(A\) apart from in rows \(j\) and \(k\). The elements in these rows are calculated

\[
GA_{j,i} = ca_{j,i} + sa_{k,i}, \quad GA_{k,i} = -sa_{j,i} + ca_{k,i}
\]

(28)

4.2 The QR Algorithm

The implementation of the QR algorithm for a complex symmetric matrix, \(A\), that is presented in appendix A.7 consists of these steps;

Algorithm 1, the QR algorithm.

1. Use similarity transformations to reduce \(A\) to tridiagonal form, \(B_1\)
2. for \(k = 1, 2, 3\ldots\)
   
   QR decompose \(B_k\), \(QR = B_k\)
   
   \(B_{k+1} = RQ\)
3. Repeat from 2.
Hessenberg reduction

The first step is known as Hessenberg reduction. An upper Hessenberg matrix is a matrix with only zeroes below the subdiagonal. Likewise, a matrix with only zeroes above the superdiagonal is lower Hessenberg. A tridiagonal matrix is both upper and lower Hessenberg. The reduction to Hessenberg form is done iteratively with Householder reflections. Assuming the rows and columns of $A$ are tridiagonal up to row $j$, $U_j$ is then chosen to be the matrix

$$U_j = \begin{pmatrix} I & 0^T \\ 0 & H_j \end{pmatrix}$$

with $I$ as the $j \times j$ identity matrix, $0$ is a $j \times (n-j)$ block of zeroes, and $H_j$ is a Householder matrix with $x = (a_{j+1,j}, a_{j+2,j}...a_{n,j})^T$. Equation (27) then gives the required $v$. The matrix $U_j$ preserves the tridiagonal structure of the top of the matrix, and transforms the $j$th column vector to $(a_1,j, a_j,...a_{j+1,j}, 0, 0,..., 0)^T$. Due to the symmetry of $A$ and $U_j$ ($U_j$, being complex symmetric and orthogonal, has the following nice properties; $U_j = U_j^T, U_j^2 = I$), the right-multiplication of $A$ with $U_j$ will zero all elements of the $j$th row that are above the superdiagonal. $U_{j+1}$ is then defined from the matrix $U_jAU_j$ (the vector $x$ used to generate $U_{j+1}$ consists of the elements below the diagonal of the $j+1$th column vector of $U_jAU_j$). $A$ is reduced to symmetric tridiagonal form after $n-2$ iterations of this similarity transformation.

$$B_1 = U_{n-2}...U_3U_2U_1AU_1U_2U_3...U_{n-2} = U^{-1}AU, \quad U = U_1U_2U_3...U_{n-2}$$

QR decomposition of $B_k$

We now wish to find a complex orthogonal matrix $Q$ and an upper triangular matrix $R$ so that $B_k = QR$. For a tridiagonal matrix, this process is done by Givens rotations, iterating along the subdiagonal, finding $c$ and $s$ for $G(s,c,j,j+1)$ so that the element on the subdiagonal is deleted and then repeating the procedure on the next element on the subdiagonal in the new matrix $G(s,c,j,j+1)B_k$. Using (28), the following system has to be solved

$$\begin{cases} -sb_{j,j} + cb_{j+1,j} = 0 \\ c^2 + s^2 = 1 \end{cases}$$

(31)
This yields
\[ c = \frac{b_{j,j}}{\sqrt{b_{j,j}^2 + b_{j+1,j}^2}}, \quad s = \frac{b_{j+1,j}}{\sqrt{b_{j,j}^2 + b_{j+1,j}^2}} \]

The algorithm for QR decomposition via Givens rotations for a tridiagonal matrix can now be stated:

\[
\begin{align*}
R &= B_k, Q = I \\
\text{for } j &= 1, 2, \ldots, n-1 \\
R &= G(b_{j+1,j}/\sqrt{b_{j,j}^2 + b_{j+1,j}^2}, b_{j,j}/\sqrt{b_{j,j}^2 + b_{j+1,j}^2}; j, j + 1)R \\
Q &= QG(b_{j+1,j}/\sqrt{b_{j,j}^2 + b_{j+1,j}^2}, b_{j,j}/\sqrt{b_{j,j}^2 + b_{j+1,j}^2}; j, j + 1)^T \\
\end{align*}
\]

Which gives
\[ R = G_{n-1}G_3G_2G_1B_k \]

and
\[ Q = (G_{n-1}G_3G_2G_1)^T = G_1^TG_2^TG_3^T\ldots G_{n-1}^T \]

This algorithm is based on the assumption that \( B_k \) is tridiagonal. For the algorithm to work repeatedly, it is required that the QR algorithm preserves the tridiagonal structure.

**Theorem 2.** If \( B_k \) is a symmetric tridiagonal matrix, then \( B_{k+1} \) is as well.

*Proof.* By calculating the transpose of the decomposition of \( B_{k+1} \) in terms of \( Q \) and \( B_k \), we conclude that \( B_{k+1} \) is symmetric.

\[
B_{k+1} = (RQ)^T = (Q^TR_k)Q = Q^T B_k^T Q = Q^T B_k Q = RQ = B_{k+1}
\]

Now, using symmetry, the structure of \( B_{k+1} \) is studied in terms of the products of the Givens rotation matrices.

\[
B_{k+1} = RQ = (RQ)^T = Q^T R^T = G_{n-1}G_3G_2G_1 R^T
\]

Since \( R \) is upper triangular, \( R^T \) is lower triangular. Assume now that \( S = G_k\ldots G_3G_2G_1R^T \) only has zeroes above the superdiagonal in columns 1 to \( k + 1 \), and only has zeroes above the diagonal in columns \( k + 2 \) to \( n \).

The elements of \( G_{k+1}S \) that are different from those of \( S \) are located in rows \( k + 1 \) and \( k + 2 \). These are calculated from (28)

\[
(G_{k+1}S)_{k+1,i} = ca_{k+1,i} + sa_{k+2,i}, \quad (G_{k+1}S)_{k+2,i} = -sa_{k+1,i} + ca_{k+2,i}
\]
The elements in rows $k+1$ and $k+2$ of $G_{k+1}S$ that are above the superdiagonal are then zero as, by assumption, $a_{k+1,i} = 0$ and $a_{k+2,i} = 0$ for $i \geq k + 3$. This means that iteratively multiplying $R^T$ with the $G_k$s transforms it into a lower Hessenburg matrix. As this matrix is also symmetric, it must have zeroes below the subdiagonal, and is therefore tridiagonal.

The final theorem required for the use of the QR algorithm is stated without proof,

**Theorem 3.** If $A \in \mathbb{C}^{n \times n}$ is complex symmetric, with $n$ eigenvalues with distinct moduli, then as $k \to \infty$, $B_k$ converges to a diagonal matrix, with the eigenvalues of $A$ on the diagonal.

For proof of the formulation of Theorem 3 with regards to unitary decomposition instead of complex orthogonal, see [3] or [4]. It is worth reiterating that this algorithm is not guaranteed to work for all complex symmetric matrices, as the the complex symmetric structure does not necessarily imply that the eigenvalues of $A$ have the properties required for Theorem 3 to hold. Also of note is that breakdowns may occur in this particular implementation, in particular, the fact that $\langle x, x \rangle_T = 0 \Rightarrow x = 0$, can be pathological in, for example, the calculations of Hessenberg matrices. This version of the algorithm is, in general, not used for calculating the eigenvalues of matrices with $n \geq 25$ as, while accurate, it is not competitive with respect to time. It is common to use so-called “shifts” to accelerate the convergence of $B_k$, i.e. compute the QR decomposition of $B_k - \mu I$ instead of $B_k$, for some well-chosen $\mu$. This is not used here as there is a desire to preserve approximations of the eigenvectors of $A$ (which are provided by the column vectors of the matrix $UQ_1Q_2\ldots Q_k$). The column vectors of the $Q$s generated by the QR decomposition of the shifted matrix do not provide the eigenvectors in the same way. Despite its shortcomings, it is not without justification that this version was used, as is seen in the next section.

### 4.3 The Jacobi-Davidson method

When calculating the resonance energies, most of the spectrum of $\hat{H}(\eta)$ is not of interest. The QR algorithm produces approximations for all eigenvalues of $\hat{H}(\eta)$, at great computational cost. Here a method to approximate a single eigenvalue that is closest to a given value, $\tau$, is presented. The idea is to approximate the eigenvalue in a subspace $\mathcal{U} \subset \mathbb{C}^n$ ($\mathcal{U}$ is called the
search space), \( \dim(U) \ll n \), and iteratively add suitable vectors to \( U \) to extend the search space until the approximation is good enough, i.e. it fulfills some convergence criterion. These two steps are known as extraction and expansion.

**Extraction**

Let \( \dim(U) = k \), and \( U \) be an \( n \times k \) matrix whose column vectors form an orthonormal basis (with respect to \( \langle \cdot, \cdot \rangle_T \)). The desire is to, for a complex symmetric matrix \( A \), calculate an approximate eigenpair to \( \tau, (u, \phi) \), \( u \in U \), such that the error, or residual, \( r \),

\[
r := Au - \phi u
\]

is orthogonal to \( U \). Equation (32) is known as the Galerkin, or Ritz-Galerkin, condition. The convergence criterion is based on the norm of \( r \) with regard to the standard inner product \( \|r\| = \sqrt{r^T r} \), i.e. the process terminates when \( \|r\| < \epsilon \). The orthogonality condition can be written as

\[
U^T r = 0 \iff U^T (A - \phi I) u = 0
\]

Letting \( u = Uc, c \in \mathbb{C}^k \), this is reduced to the \( k \)-dimensional eigenproblem

\[
U^T A U c = \phi c
\]

and the pair \( (c, \phi) \) with \( \phi \) selected from the spectrum of \( U^T A U \) so that \( |\tau - \phi| \) is minimized. Here it is worth noting that \( U^T A U \) is a complex symmetric matrix. The approximate eigenpair \( (Uc, \phi) \) is known as a “Ritz pair”. To be able to check the convergence criterion, this is normalized, \( u = Uc/c^T c \).

Now the so-called Rayleigh quotient for a complex symmetric matrix \( A \), and a vector \( v \), is introduced. It is defined as

\[
R(A, v) := \frac{v^T A v}{v^T v}
\]

For an eigenvector, \( a_i \) of \( A \), the Rayleigh quotient returns the eigenvalue \( \lambda_i \) corresponding to \( A \). Using \( r \perp U \Rightarrow (u, r)_T = 0 \), it is now seen that \( \phi = R(A, u) \).

\[
R(A, u) = \frac{u^T A u}{u^T u} = \frac{u^T (\phi u + r)}{u^T u} = \phi
\]

This shall be used later on when discussing convergence of the algorithm.
Expansion

Let $\lambda$ be the eigenvalue of $A$ that is closest to $\tau$. Now, given an approximate eigenpair $(u, \phi)$, $(u, u)^T = 1$, and assuming the convergence criterion is not met, a suitable vector is to be chosen to expand the search space. The Jacobi-Davidson method is to find a vector $s$, so that

$$A(u + s) = \lambda(u + s)$$

and also $(s, u)^T = 0$. Equation (36) is rewritten as

$$(A - \lambda I)s = (\lambda I - A)u$$

As $\lambda$ is unknown at this stage, this cannot be solved completely. Instead, $\lambda$ is approximated with $R(A, u)$, and solved in the subspace $u^{+T}$. The projector $I - uu^T$ is used. Applying the projector to the right hand side of the equation gives

$$(I - uu^T)(\lambda I - A)u = -Au + u(u^TAu) = -(Au - \phi u) = -r$$

The condition $(s, u) = 0$ can be formulated as

$$(I - uu^T)s = s$$

Equation 37 now reads

$$(I - uu^T)(A - R(A, u)I)(I - uu^T)s = -r$$

Equation 40 is called the Jacobi-Davidson correction equation. The system is solved for $s$, which is then added to the search space, and the process is repeated. The algorithm can now be formulated

Algorithm 2, the Jacobi Davidson algorithm.

1. Given $\tau$ and $\epsilon$, choose an initial vector, $b$. Let $s = b$
2. for $k=1,2,\ldots$
3. Use the complex orthogonal Gram-Schmidt method on $\{e_1, e_2, \ldots, e_{k-1}, s\}$, with $\{e_i\}$ being the column vectors of $U_{k-1}$
4. Let the columns of $U_k$ consist of the vectors produced in step 3.
5. Find the eigenpair $(c, \phi)$ of $U_k^T A U_k$ that minimizes $|\tau - \phi|$
6. Calculate $u = U_k c / (U_k c)^T (U_k c)$
7. $r$ is calculated with regard to the standard norm of $u$, $r = \frac{(A - \phi I)u}{\|u\|}$
8. if $\|r\| < \epsilon$, stop.
10. Repeat from 2.
The complex symmetric QR algorithm is used in step 5 to provide the Ritz pair, \((c, \phi)\). In the implementation of the algorithm in the appendix (section A.8), a variation of the algorithm is used that prevents the dimension of \(U\) growing too large. This is done by inserting a new step, 8b. This is a restart, replacing \(U\) with \(u\), and resetting \(k\). In this way the use of the implementation of the QR algorithm presented in section 4.2 is justifiable, as the combination of the fact that accurate approximations of both the eigenvalue and eigenvector are required, and that the algorithm works well for matrices of the size that it deals with in the JD algorithm.

8b. if \(k > 25\), then \(U_k = u\), \(k = 1\)

Convergence of the Jacobi-Davidson method

Motivation of the convergence of Jacobi-Davidson algorithm is based on the convergence of another algorithm, the Rayleigh Quotient Iteration. This method requires an initial guess of an eigenpair \((u_1, \phi_1)\), and new approximations are created iteratively

\[
\phi_{k+1} = R(A, u_k), \quad \hat{u}_{k+1} = (A - \phi_k I)^{-1} u_k, \quad u_{k+1} = \frac{\hat{u}_{k+1}}{\sqrt{(\hat{u}_{k+1}, u_{k+1})}} 
\] (41)

The method breaks down if \((A - \phi_k I)\) becomes singular, in which case the eigenvalue has been found, or if \((\hat{u}_{k+1}, \hat{u}_{k+1})_T = 0\), in which case the RQI must be restarted with a different initial vector. The following theorem concerns the local convergence of the RQI

Theorem 4. Assume \(u_k \to x\), as \(k \to \infty\), \(Ax = \lambda x\). Then
i) \(u_k\) can be written as \(u_k = \alpha_k (x + \delta_k d_k)\), \((x, d_k)_T = 0\), \((d_k, d_k)_T = 1\).
ii) \(\phi_k \to \lambda\).
iii) The local convergence of \(u_k\) to \(x\) is cubic, that is

\[
\delta_{k+1} = \mathcal{O}(\delta_k^3) 
\] (42)

Sketch of Proof, as presented in [5]
i) To start with, it is noted that

\[
u_k = xx^T u_k + (I - xx^T)u_k
\]
and that \( \langle x, (I - xx^T)u \rangle_T = 0 \). The choices for the variables appear after normalization

\[
\alpha_k = x^Tu_k, \quad d_k = \frac{(I - xx^T)u_k}{\sqrt{u_k^T(I - xx^T)u_k}}, \quad \delta_k = \frac{\sqrt{u_k^T(I - xx^T)u_k}}{x^Tu_k}
\]

ii) Using the Rayleigh Quotient, and noting that as \( u_k \to x \), \( \delta_k \to 0 \), the convergence of \( \phi_k \) can now be proved

\[
\phi_k = u_k^T A u_k = \alpha_k^2(x^T + \delta_k d_k^T)(\lambda x + \delta_k A d_k) = \alpha_k^2(\lambda + \delta_k^2 d_k^T A d_k) = \frac{1}{1 + \delta_k^2} (\lambda + \delta_k^2 d_k^T A d_k)
\]

(43)

For the third equality of (43) to hold it is essential that \( A \) is complex symmetric, so as to guarantee that \( x^T A d = 0 \). Equation (43) is now rewritten

\[
\lambda - \phi_k = \frac{1}{1 + \delta_k^2} ((1 + \delta_k^2)\lambda - (\lambda + \delta_k^2 d_k^T A d_k)) = \frac{\delta_k^2}{1 + \delta_k^2} d_k^T (\lambda I - A) d_k
\]

(44)

Using series expansion of \( \frac{\delta_k^2}{1 + \delta_k^2} \), and assuming an upper bound for \( |d_k^T(\lambda I - A) d_k| \) gives

\[
|\lambda - \phi_k| = |\delta_k^2 d_k^T (\lambda I - A) d_k| + O(\delta_k^4) = O(\delta_k^2)
\]

(45)

This proves the convergence of \( \phi_k \). The final part of the proof requires the calculation of \( u_{k+1} \) in terms of \( u_k \). To start with

\[
(\lambda - \phi_k) u_k = \alpha_k ((A - \phi_k I)x + \delta_k (\lambda - \phi_k) d_k)
\]

(46)

Then

\[
u_{k+1} = s(A - \phi_k I)^{-1} u_k = \frac{s \alpha_k}{\lambda - \phi_k} (x + \delta_k (\lambda - \phi_k) (A - \phi I)^{-1} d_k)
\]

(47)

The term \( s \) is normalizing, ensuring that \( \langle u_{k+1}, u_{k+1} \rangle_T = 1 \). The complex symmetry of \( A \) is again used, ensuring that \( (A - \phi_k I)^{-1} \) is also symmetric. This is then used to conclude that

\[
\langle x, (A - \phi_k I)^{-1} d_k \rangle_T = \langle d_k, (A - \phi_k I)^{-1} x \rangle_T = \frac{1}{\lambda - \phi_k} \langle x, d_k \rangle_T = 0
\]

(48)
Since the requirement $\langle x, d_{k+1} \rangle_T = 0$ is met for $d_{k+1} = \gamma (A - \phi_k I)^{-1} d_k$, $\gamma$ being a normalizing factor, it can now be concluded that $\alpha_{k+1} = \frac{\delta_{k+1}}{\lambda_k - \phi}$, and hence

$$u_{k+1} = \alpha_{k+1} (x + \delta_k (\lambda - \phi_k)(A - \phi_k I)^{-1} d_k)$$ (49)

This gives the relation between $\delta_{k+1}d_{k+1}$ and $\delta_k d_k$

$$\delta_{k+1}d_{k+1} = \delta_k (\lambda - \phi_k)(A - \phi_k I)^{-1} d_k$$

Assuming $d_k^T (A - \phi_k I)^{-2} d_k$ is bounded, (45) provides the required order,

$$\delta_{k+1} = \frac{1}{\gamma} \delta_k (\lambda - \phi_k) = O(\delta_k^3), \quad \gamma = \frac{1}{\sqrt{d_k^T (A - \phi_k I)^{-2} d_k}}$$ (50)

Returning now to the Jacobi-Davidson method, (40) provides the link between the two methods, supposing $k$ iterations of the JD algorithm have been performed, (40) reads

$$(I - u_k u_k^T)(A - \phi_k I)(I - u_k u_k^T)s = (A - \phi_k I)u_k$$ (51)

The explicit solution to this equation is $s = -u_k + \beta (A - \phi_k I)^{-1} u_k$, $
\beta = \frac{1}{u_k (A - \phi_k I)^{-1} u_k}$, as

$$(I - u_k u_k^T)(-u_k + \beta (A - \phi_k I)^{-1} u_k) = \beta (A - \phi_k I)^{-1} u_k - u_k u_k^T (\beta (A - \phi_k I)^{-1} u_k) = \beta (A - \phi_k I)^{-1} u_k - u_k$$

And

$$(A - \phi_k I)(\beta (A - \phi_k I)^{-1} u_k - u_k) = \beta u_k - (A - \phi_k I) u_k$$

Finally

$$(I - u_k u_k^T)(\beta u_k - (A - \phi_k I) u_k) = \beta (u_k - u_k u_k^T u_k) - ((A - \phi_k I) u_k - u_k u_k^T (A - \phi_k I) u_k) = -r + u_k u_k^T r = -r + \langle u_k, r \rangle_T u_k = -r$$

When $s$ is added to $U$, the vector $-u_k + \beta (A - \phi_k I)^{-1} u_k$ extends the search space. Since $u_k \in U$, this is equivalent to extending the space with just
the vector \((A - \phi_k I)^{-1}u_k\). This is the next vector in the Rayleigh Quotient Iteration, so the Jacobi-Davidson algorithm can be seen as a version of the RQI, where the previous iterations are stored. Since RQI converges, JD will also converge. By creating the search space, the algorithm can create a next estimate that is more accurate than just \((A - \phi_k I)^{-1}u_k\). This is called subspace acceleration. Another advantage the JD method has over RQI, is that as \(\phi_k\) approaches \(\lambda\), \((A - \lambda I)\) gets closer to being singular. This causes problems when solving the RQI equation, \((A - \lambda I)u_{k+1} = u_k\), numerically as the matrix becomes ill-conditioned, which can give inaccurate results. This can cause convergence problems with RQI, but is not necessarily a critical problem to JD, as the previous iterations have been saved, and can preserve the accuracy of the estimate. It is also worth noting that there are variations of the algorithm to improve convergence when calculating interior eigenvalues, two alternatives are presented in [5]; Harmonic Ritz vectors and Refined Ritz vectors. The algorithm in appendix A.8, while still not competitive compared with Matlab’s inbuilt function eigs, still appears to function as desired, it succeeded at calculating the first resonance energies for the CAP in the rather extreme case \(n = 2000, L = 32, \eta = 0.011\). Other methods of improving the performance of the algorithm include finding an approximate solution to (40), as solving this system exactly is one of the more computationally demanding aspects of the algorithm. This is called the inexact Jacobi-Davidson algorithm. A full discussion of the Jacobi-Davidson algorithm, its adaptation to the complex symmetric case, and its variants is presented in [5] and [6].

**Acknowledgements**

I thank Michael Melgaard for providing the topic of the project, as well as for his supervision and assistance with the realization of it.
A Matlab code

A.1 An implementation of Newton’s method to calculate the resonance energies

```matlab
V=10;
a=1;
E= %See Table 1 for the used initial values of E.
for(n =1:100)
    k1=sqrt(2*(E+V));
    k2=sqrt(2*(E-V));
    k3=sqrt(2*E);
    T1 = k2*k3*sin(k1*a)*cos(k2*a);
    T2 = k1*k3*sin(k2*a)*cos(k1*a);
    T3 = 1i*k1*k2*cos(k1*a)*cos(k2*a);
    T4 = -1i*(k2^2)*sin(k1*a)*sin(k2*a);
    dk1 = 1/k1;
    dk2 = 1/k2;
    dk3 = 1/k3;
    dT1 = dk2*k3*sin(k1*a)*cos(k2*a)+...
          dk3*k2*sin(k1*a)*cos(k2*a)+...
          k2*k3*cos(k1*a)*cos(k2*a)*a*dk1+...
          k2*k3*sin(k1*a)*sin(k2*a)*a*dk2;
    dT2 = dk1*k3*sin(k2*a)*cos(k1*a)+...
          dk3*k1*sin(k2*a)*cos(k1*a)+...
          k1*k3*cos(k2*a)*a*dk2*cos(k1*a)+...
          k1*k3*sin(k2*a)*sin(k1*a)*a*dk1;
    dT3 = 1i*dk1*k2*cos(k1*a)*cos(k2*a)+...
          1i*dk2*cos(k1*a)*cos(k2*a)+...
          1i*k1*k2*sin(k1*a)*a*dk1*cos(k2*a)+...
          1i*k1*k2*cos(k1*a)*sin(k2*a)*a*dk2;
    dT4 = -1i*2*dk2*sin(k1*a)*sin(k2*a)+...
          -1i*(k2^2)*cos(k1*a)*a*dk1*sin(k2*a)+...
          -1i*(k2^2)*sin(k1*a)*cos(k2*a)*a*dk2;
    E = E-(T1+T2+T3+T4)/(dT1+dT2+dT3+dT4)
end
E
```
A.2 A program that calculates the eigenvalues of $\hat{H}$ using the Complex Symmetric QR algorithm.

```matlab
N=50; L=4; a=1; V=10; eta=0.32;
H=zeros(N);
for j=1:N-1
    for k=j+1:N
        T1=(2*V)/(pi*(j+k))*sin(pi*a*(j+k)/L);
        T2=-(2*V)/(pi*(k-j))*sin(pi*a*(k-j)/L);
        T3=(V/(pi*(k-j)))*sin(2*pi*a*(k-j)/L);
        T4=-(V/(pi*(k+j)))*sin(2*pi*a*(k+j)/L);
        T5=-1i*eta*(-1)^(k-j)*(2*L*(L-2*a))/(pi^2*(k-j)^2);
        T6=1i*eta*(-1)^(k+j)*(2*L*(L-2*a))/(pi^2*(k+j)^2);
        T7=-1i*eta*(2*L^2/(pi^3*(k-j)^3))*sin(2*pi*(k-j)*a/L);
        T8=1i*eta*(2*L^2/(pi^3*(k+j)^3))*sin(2*pi*(k+j)*a/L);
        H(j,k)=T1+T2+T3+T4+T5+T6+T7+T8;
        H(k,j)=H(j,k);
    end
end
for j=1:N
    S1 = (pi^2*j^2)/(2*L^2); S2 = V/(pi*j)*sin(2*pi*j*a/L);
    S3 = -V/(2*pi*j)*sin(4*pi*j*a/L);
    S4 = -1i*eta*(L-2*a)^3/(3*L);
    S5 = 1i*eta*(L*(L-2*a))/(2*pi^2*j^2);
    S6 = 1i*eta*(L^2)/(4*pi^2*j^2)*sin(4*pi*j*a/L);
    H(j,j)=S1+S2+S3+S4+S5+S6;
end
E=csQR(H);
```
A.3 A program that calculates the trajectory of the resonance energy near $13.8 - 1.27i$ for $N = 504, L = 12$, using the Complex Symmetric Jacobi-Davidson algorithm

```matlab
N=504; L=12; a=1; V=10; H=zeros(N); x = zeros(300,1);
for m = 1:301
    eta = 0.005*(1.1^((m-1)/3));
    for j=1:N-1
        for k=j+1:N
            T1=(2*V)/(pi*(j+k))*sin(pi*a*(j+k)/L);
            T2=-(2*V)/(pi*(k-j))*sin(pi*a*(k-j)/L);
            T3=(V/(pi*(k-j)))*sin(2*pi*a*(k-j)/L);
            T4=-(V/(pi*(k+j)))*sin(2*pi*a*(k+j)/L);
            T5=-1i*eta*(-1)^(k-j)*(2*L*(L-2*a))/(pi^2*(k-j)^2);
            T6=1i*eta*(-1)^(k+j)*(2*L*(L-2*a))/(pi^2*(k+j)^2);
            T7=-1i*eta*(2*L^2/((pi^3*(k-j)^3))*sin(2*pi*(k-j)*a/L);
            T8=1i*eta*(2*L^2/((pi^3*(k+j)^3))*sin(2*pi*(k+j)*a/L);
            H(j,k)=T1+T2+T3+T4+T5+T6+T7+T8;
            H(k,j)=H(j,k);
        end
    end
    for j=1:N
        S1 = (pi^2*j^2)/(2*L^2);
        S2 = V/(pi*j)*sin(4*pi*j+a/L);
        S3 = -V/(2*pi*j)*sin(4*pi*j+a/L);
        S4 = -1i*eta*(L-2*a)^3/(3*L);
        S5 = 1i*eta*(L*(L-2*a)/(2*pi^2*j^2));
        S6 = 1i*eta*(L^2)/(4*pi^2*j^2)*sin(4*pi*j*a/L);
        H(j,j)=S1+S2+S3+S4+S5+S6;
    end
    x(m)=JDCS(H,13.8-1.27i,0.00001);
end
```
A.4 A program that uses Householder reflections to transform a complex symmetric matrix to symmetric tridiagonal form.

```matlab
function [B, Q] = cstridiag(A)
B = A;
[n, n] = size(B);
Q = eye(n, n);
for j = 1:n-2
    x = B(j+1:end, j);
    k = sqrt((x.')*x);
    if (x(1) == sqrt((x.')*x)) || (x(1) == -sqrt((x.')*x))
        k = -x(1);
    end
    [s, ~] = size(x);
    v = (1/(sqrt(2*((x.')*x-k*x(1))))) * (x-k*eye(s, 1));
    H = eye(s, s) - 2*v*(v.');
    P = zeros(n, n);
    P(1:j, 1:j) = eye(j, j);
    P(j+1:end, j+1:end) = H;
    B = P*B*(P.');
    Q = Q*(P.');
    B(j+2:end, j) = 0;
    B(j, j+2:end) = 0;
end
```

A.5 The complex symmetric Gram-Schmidt algorithm

```matlab
function Q = CSGS(A)
[m, n] = size(A); Q = zeros(m, n); R = zeros(n, n);
for j = 1:n
    v = A(:, j);
    for k = 1:j-1
        R(k, j) = Q(:, k)'*A(:, j); v = v - R(k, j)*Q(:, k);
    end
    R(j, j) = sqrt(v.'*v); Q(:, j) = v/R(j, j);
end
```
A.6 A program that uses Givens Rotations to calculate the QR factorization of a symmetric tridiagonal matrix.

```matlab
function [Q, R] = givrotsQR(A)
[n, n]=size(A);
R=A;
Q=eye(n,n);
for j =1:n-1;
    G=eye(n,n);
    if R(j+1,j) \neq 0;
        a=R(j,j);
        b=R(j+1,j);
        r=sqrt(a^2+b^2);
        c=a/r;
        s=b/r;
        G(j,j)=c;
        G(j,j+1)=s;
        G(j+1,j)=-s;
        G(j+1,j+1)=c;
    end
    Q=Q*(G.');
    R=G*R;
end
for l=2:n
    for m =1:l-1
        R(l,m)=0;
    end
end
```

A.7 The complex symmetric QR algorithm

```matlab
function [E V] = csQR(A, epsilon)
[B Q1] = cstridiag(A);
[n,1] = size(B);
E = zeros(n,1);
Q2 = eye(n,n);
J = zeros(n-1,1);
for k = 1:5000
    [Q R] = givrotsQR(B);
    B = R*Q;
    Q2 = Q2*Q;
    for m = 1:n-1
        J(m) = abs(B(m+1,m));
    end
    s = max(J);
    if s < epsilon
        E = diag(B);
        break
    end
    E = sort(diag(B));
end
V = Q1*Q2;
```

A.8 The complex symmetric Jacobi-Davidson algorithm

```matlab
function lambda = JDCS(A,sigma,epsilon)

[n, n] =size(A);

u1=rand(n,1)+1i*rand(n,1);
u1=u1/sqrt(((u1.')*u1));
s=u1;
U=s;
k=1;
while k < 25

Q=zeros(n,k);
if k ~= 1
    Q(1:n,1:k-1)=U;
    Q(1:n,k)=s;
    U=CSGS(Q);
end
k=k+1;
R=(U.')*(A*U);
[E B]=csQR(R,0.00000001);
[~, I]=min(abs(E-sigma));
phi=E(I);
c=B(:,I);
c=c/sqrt((c.')*c);
u=U*c;
r=((A-phi*eye(n))*u)/norm(u);
lambda=phi;
if k > 24
    U=u;
k=1;
end
if norm(r) < epsilon
    break;
end

t=(eye(n,n)-u*(u.'))*(A-phi*eye(n,n))*(eye(n,n)-u*(u.'));
s=linsolve(t,(-r));
end
```
B Computation of elements of $\hat{H}$

To simplify calculations, the integral is split up;

$$\hat{H}(\eta)_{j,k} = \int_0^\infty \left( -\frac{1}{2} \frac{d^2}{dr^2} \phi_k(r) \right) \phi_j(r) dr + \int_0^\infty V(r) \phi_k(r) \phi_j(r) dr - i\eta \int_0^\infty \hat{W}(r) \phi_k(r) \phi_j(r) dr$$

using $-\frac{d^2}{dr^2} \sin(xr) = x^2 \sin(xr)$ gives

$$\int_0^\infty \left( -\frac{1}{2} \frac{d^2}{dr^2} \phi_k(r) \right) \phi_j(r) dr = \frac{\pi^2 k^2}{2L^2} \int_0^\infty \phi_k(r) \phi_j(r) dr = \frac{\pi^2 k^2}{2L^2} \delta_{jk}$$

The other integrals are somewhat more complicated;

$$\int_0^\infty V(r) \phi_k(r) \phi_j(r) dr = \int_0^a -V_0 \phi_k(r) \phi_j(r) dr + \int_a^{2a} V_0 \phi_k(r) \phi_j(r) dr$$

(if $j \neq k$)

$$= -V_0 \frac{1}{L} \left[ \frac{L}{\pi(k-j)} \sin \left( \frac{\pi(k-j)r}{L} \right) - \frac{L}{\pi(k+j)} \sin \left( \frac{\pi(k+j)r}{L} \right) \right]_0^a$$

$$V_0 \frac{1}{L} \left[ \frac{L}{\pi(k-j)} \sin \left( \frac{\pi(k-j)r}{L} \right) - \frac{L}{\pi(k+j)} \sin \left( \frac{\pi(k+j)r}{L} \right) \right]_a^{2a}$$

$$= \frac{V_0}{\pi(k+j)} \sin \left( \frac{\pi(k+j)a}{L} \right) - \frac{V_0}{\pi(k-j)} \sin \left( \frac{\pi(k-j)a}{L} \right) +$$

$$\frac{V_0}{\pi(k-j)} \sin \left( \frac{2\pi(k-j)a}{L} \right) - \frac{V_0}{\pi(k+j)} \sin \left( \frac{2\pi(k+j)a}{L} \right) +$$

$$\frac{V_0}{\pi(k-j)} \sin \left( \frac{\pi(k-j)a}{L} \right) + \frac{V_0}{\pi(k+j)} \sin \left( \frac{\pi(k+j)a}{L} \right)$$

$$= \frac{2V_0}{\pi(k+j)} \sin \left( \frac{\pi(k+j)a}{L} \right) - \frac{2V_0}{\pi(k-j)} \sin \left( \frac{\pi(k-j)a}{L} \right) +$$

$$\frac{V_0}{\pi(k-j)} \sin \left( \frac{2\pi(k-j)a}{L} \right) - \frac{V_0}{\pi(k+j)} \sin \left( \frac{2\pi(k+j)a}{L} \right)$$
if \( j = k; \)

\[
    = -V_0 \frac{1}{L} \left[ r - \frac{L}{2\pi k} \sin \left( \frac{2\pi kr}{L} \right) \right]_0^a + V_0 \frac{1}{L} \left[ r - \frac{L}{2\pi k} \sin \left( \frac{2\pi kr}{L} \right) \right]_a^{2a}
\]

\[
    = \frac{V_0}{\pi k} \sin \left( \frac{2\pi ka}{L} \right) - \frac{V_0}{2\pi k} \sin \left( \frac{4\pi ka}{L} \right)
\]

The final integral requires repeated integration by parts;

\[
    \int_0^\infty \hat{W}(r) \phi_k(r) \phi_j(r) dr = \int_{2a}^L (r - 2a)^2 \phi_k(r) \phi_j(r) dr
\]

\[
    = \frac{2}{L} \int_{2a}^L (r - 2a)^2 \sin \left( \frac{\pi kr}{L} \right) \sin \left( \frac{\pi jr}{L} \right) dr
\]

\[
    = \frac{1}{L} \int_{2a}^L (r - 2a)^2 \cos \left( \frac{\pi (k - j)r}{L} \right) - (r - 2a)^2 \cos \left( \frac{\pi (k + j)r}{L} \right) dr
\]

\( j \neq k \) gives

\[
    = \frac{1}{L} \left[ (r - 2a)^2 \frac{L}{\pi(k-j)} \sin \left( \frac{\pi(k-j)r}{L} \right) - (r - 2a)^2 \frac{L}{\pi(k+j)} \sin \left( \frac{\pi(k+j)r}{L} \right) \right]_{2a}^L + \]

\[
    - \frac{1}{L} \int_{2a}^L 2(r - 2a) \frac{L}{\pi(k-j)} \sin \left( \frac{\pi(k-j)r}{L} \right) - 2(r - 2a) \frac{L}{\pi(k+j)} \sin \left( \frac{\pi(k+j)r}{L} \right) dr
\]

\[
    = \frac{1}{L} \left[ 2(r - 2a) \frac{L^2}{\pi^2(k-j)^2} \cos \left( \frac{\pi(k-j)r}{L} \right) - 2(r - 2a) \frac{L^2}{\pi^2(k+j)^2} \cos \left( \frac{\pi(k+j)r}{L} \right) \right]_{2a}^L + \]

\[
    - \frac{1}{L} \int_{2a}^L \frac{2L^2}{\pi^2(k-j)^2} \cos \left( \frac{\pi(k-j)r}{L} \right) - \frac{2L^2}{\pi^2(k+j)^2} \cos \left( \frac{\pi(k+j)r}{L} \right) dr
\]

\[
    = \left( \frac{2L(L-2a)}{\pi^2(k-j)^2} (-1)^{k-j} - \frac{2L(L-2a)}{\pi^2(k+j)^2} (-1)^{k+j} \right) + \]

\[
    - \left[ \frac{2L^2}{\pi^3(k-j)^3} \sin \left( \frac{\pi(k-j)r}{L} \right) - \frac{2L^2}{\pi^3(k+j)^3} \sin \left( \frac{\pi(k+j)r}{L} \right) \right]_{2a}^L
\]

37
\[
\left( \frac{2L^2}{\pi^3(k-j)^3} \sin \left( \frac{2\pi(k-j)a}{L} \right) \right) = \left( \frac{2L(L-2a)}{\pi^2(k-j)^2} (-1)^{k-j} - \frac{2L(L-2a)}{\pi^2(k+j)^2} (-1)^{k+j} \right) + \left( \frac{2L^2}{\pi^3(k+j)^3} \sin \left( \frac{2\pi(k+j)a}{L} \right) \right)
\]

\(k = j\) gives

\[
\frac{1}{L} \int_{2a}^{L} (r - 2a)^2 - (r - 2a)^2 \cos \left( \frac{2\pi kr}{L} \right) \, dr
\]

\[
= \frac{(L - 2a)^3}{3L} - \frac{1}{L} \left[ (r - 2a)^2 \frac{L}{2\pi k} \sin \left( \frac{2\pi kr}{L} \right) \right]_{2a}^{L} - \frac{1}{L} \int_{2a}^{L} 2(r - 2a) \frac{L}{2\pi k} \sin \left( \frac{2\pi kr}{L} \right) \, dr
\]

\[
= \frac{(L - 2a)^3}{3L} - \frac{2L(L - 2a)}{4\pi^2 k^2} \cos \left( \frac{2\pi kr}{L} \right) - \frac{2L^2}{4\pi^2 k^2} \cos \left( \frac{2\pi kr}{L} \right) \, dr
\]

\[
= \frac{(L - 2a)^3}{3L} - \frac{2L(L - 2a)}{4\pi^2 k^2} - \frac{L^2}{4\pi^3 k^3} \sin \left( \frac{4\pi ka}{L} \right)
\]

The non-diagonal entries are therefore

\[
\hat{H}(\eta)_{j,k} = \frac{2V_0}{\pi(k+j)} \sin \left( \frac{\pi(k+j)a}{L} \right) - \frac{2V_0}{\pi(k-j)} \sin \left( \frac{\pi(k-j)a}{L} \right) + \frac{V_0}{\pi(k-j)} \sin \left( \frac{2\pi(k-j)a}{L} \right) - \frac{V_0}{\pi(k+j)} \sin \left( \frac{2\pi(k+j)a}{L} \right) + -i\eta \left( \frac{2L(L-2a)}{\pi^2(k-j)^2} (-1)^{k-j} - \frac{2L(L-2a)}{\pi^2(k+j)^2} (-1)^{k+j} \right) + -i\eta \left( \frac{2L^2}{\pi^3(k-j)^3} \sin \left( \frac{2\pi(k-j)a}{L} \right) \right) - \frac{2L^2}{\pi^3(k+j)^3} \sin \left( \frac{2\pi(k+j)a}{L} \right)
\]

And the diagonal elements

\[
\hat{H}(\eta)_{k,k} = \frac{\pi^2 k^2}{2L^2} + \frac{V_0}{\pi k} \sin \left( \frac{2\pi ka}{L} \right) - \frac{V_0}{2\pi k} \sin \left( \frac{4\pi ka}{L} \right) + -i\eta \left( \frac{(L-2a)^3}{3L} - \frac{2L(L-2a)}{4\pi^2 k^2} - \frac{L^2}{4\pi^3 k^3} \sin \left( \frac{4\pi ka}{L} \right) \right)
\]

38
References


