Centro Svizzero di Calcolo Scientifico

CSCS

Swiss Scientific Computing Center

CSCS-TR-94-04

ETH Eidgenössische Technische Hochschule Zürich
Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Swiss Federal Institute of Technology Zurich

Arnoldi Method Applied to Burgers' Equation

W. Sawyer, G. Kreiss, D. Sorensen and J. Lambers

May 10, 1994

Technical Report

Arnoldi Method Applied to Burgers' Equation

SeRD-CSCS-TR-94-04

Abstract:

The Arnoldi method for the computation of eigenvalues of large non-symmetric matrices is applied to Burgers' equation: $u_t = \left(\frac{u^2}{2}\right)_x + \varepsilon u_{xx}$. The space discretization of the problem leads to the system of equations $\mathbf{v}_t = \mathbf{F}(\mathbf{v})$. The asymptotic behavior around the steady state solution is determined by those eigenvalues of the Jacobian $\frac{\partial \mathbf{F}}{\partial \mathbf{v}}$ evaluated at a quasi-steady state $\mathbf{v}_{\mathrm{st}}$ which have the smallest negative real part.

Two variants of the Arnoldi method are used to find the desired eigenvalues: the basic algorithm supplemented by two different eigenvalue localizers: polynomial filtering and spectral transformation. The former is available in the public-domain library **ARPACK**, while the latter is problem-specific and is therefore hand-coded. Numerical results are presented and the relative performance of the two algorithms compared.

William Sawyer
CSCS–ETH, Section of Research and Development (SeRD),
Swiss Scientific Computing Center,
La Galleria, CH-6928 Manno, Switzerland
sawyer@cscs.ch
Gunilla Kreiss
Royal Institute of Technology,
Department of Computer Science,
Stockholm, Sweden
gunillak@nada.kth.se
Dan Sorensen
Department of Mathematical Sciences,
Rice University, Houston TX, 77251-1829
sorensen@rice.edu
James Lambers
Stanford University,
SCCM Program,
Stanford CA, 94305-2140
lambers@sccm.stanford.edu

Technical Report

Table of Contents

List of Figures

1 Introduction

The availability of public-domain software libraries has never been as great as today. The usefulness of this software, however, is relatively difficult to evaluate, as this software is not always tested on typical problems in the application user's view.

In this paper, we evaluate a newly available software library eigenvalue solver package, **ARPACK** [1] (available from **netlib** in the **SCALAPACK** library)[1]) on a realistic test problem. We compare the use of **ARPACK** eigenvalue solvers for a few eigenvalues of a non-symmetric system, to those of a hand-coded algorithm tailored to this problem.

We consider Burgers' equation:

$$u_t = \left(\frac{u^2}{2}\right)_x + \varepsilon u_{xx} \quad \varepsilon > 0, \quad u(0,t) = -1 \;\; u(1,t) = 1 \tag{1}$$

In order to solve this problem, this equation is discretized in space over $n = N - 1$ grid points $x_i = ih$ where $h = 1/N$ and $i = 0, 1, \ldots, N$ ($v_{t0} = -1$ and $v_{tN} = 1$), using central differences we get a system of ordinary differential equations

$$\begin{aligned} v_{t1} &= \frac{{v_2}^2 - 1}{4h} + \varepsilon\frac{v_2 - 2v_1 - 1}{h^2} \\ v_{ti} &= \frac{{v_{i+1}}^2 - {v_{i-1}}^2}{4h} + \varepsilon\frac{v_{i+1} - 2v_i + v_{i-1}}{h^2} \quad \text{for} \quad 1 < i < n \\ v_{tn} &= \frac{1 - {v_n}^2}{4h} + \varepsilon\frac{1 - 2v_n + v_{n-1}}{h^2} \quad \text{where} \quad n = N - 1 \end{aligned}$$

or formulating this more succinctly,

$$\mathbf{v}_t = \mathbf{F}(\mathbf{v}) \tag{2}$$

We expect that $\mathbf{v} = (v_1, v_2, \ldots, v_n)^T$ where $v_i(t)$ approximates $u(ih, t)$. An analysis of the discretization error will not be given here — this can be found in standard texts on Numerical Analysis such as [5].

Of particular interest is the asymptotic behavior around the steady state solution $\mathbf{v}_{\mathrm{st}}$. We therefore let $\mathbf{v} = \mathbf{v}_{\mathrm{st}} + \tilde{\mathbf{v}}$ linearize the system about v_{st},

$$\begin{aligned} \mathbf{v} &= \mathbf{v}_{\mathrm{st}} + \tilde{\mathbf{v}} \\ \frac{d\mathbf{v}}{dt} &= \mathbf{F}(\mathbf{v}) \approx \mathbf{F}(\mathbf{v}_{\mathrm{st}}) + \left.\frac{\partial \mathbf{F}}{\partial \mathbf{v}}\right|_{\mathbf{v}_{\mathrm{st}}} \tilde{\mathbf{v}} \end{aligned}$$

Since $\frac{d\mathbf{v}}{dt} = \frac{d\tilde{\mathbf{v}}}{dt}$ and $\mathbf{F}(\mathbf{v}_{\mathrm{st}}) = 0$, this becomes

$$\frac{d\tilde{\mathbf{v}}}{dt} = \left.\frac{\partial \mathbf{F}}{\partial \mathbf{v}}\right|_{\mathbf{v}_{\mathrm{st}}} \tilde{\mathbf{v}} \tag{3}$$

The solution to (3) is

[1] A wide variety of software from **netlib** can be retrieved from the server `netlib.ornl.gov`. This is best done using the facility `xnetlib`, which is installed on many UNIX systems.

2 The Steady State Solution of Burgers' equation

We consider the steady state solution to Burgers' equation (1), and integrate once we get the following equation,

$$\frac{{u_{\mathrm{st}}}^2}{2} + \varepsilon u_{\mathrm{st}_x} = C \quad u(0,t) = -1 \;\; u(1,t) = 1$$

Equation (2) can be solved by the separation of variables. The general solution for the zero-centered steady state solution is,

$$u_{\mathrm{st}} = \sqrt{C_1} \tanh \frac{C_1(x - C_2)}{2\varepsilon} \tag{8}$$

where C_1 and C_2 can be determined from the boundary conditions. Note that in this case there will be symmetry about the point $(0.5, 0)$ thus $C_2 = 0.5$ and C_1 can be determined iteratively for a given ε. The following values are accurate to 14 places:

ε	C_1
0.0125	1.00000000000000
0.025	1.00000000412231
0.05	1.00009072163678
0.1	1.01272561672732
0.2	1.12707885005682

In the general case, of course, an explicit steady state solution is not known. Therefore, as an exercise, we ignore our knowledge of the exact solution here and attempt to calculate the steady state solution by discretizing in space, as described previously, and using a time-stepping algorithm such as *Runge-Kutta 4/5* to solve (2) until the change in the grid-function values is negligible.

While this approach may seem straightforward and seems to lead to adequate results for many values of N and ε (see figure 1), certain anomalous effects can occur: "wiggles" can appear in numerically calculated steady state solution (see figure 2) if too few grid-points are used.

To explain these wiggles, we consider the semi-discretized equation,

$$\frac{dv_i}{dt} = 0 = \frac{1}{4h}({v_{i+1}}^2 - {v_{i-1}}^2) + \frac{\varepsilon}{h^2}(v_{i+1} - 2v_i + v_{i-1}) \tag{9}$$

The solution to these difference equations has wiggles if it is not monotonic increasing. Consider three contiguous grid-function values v_{i-1}, v_i and v_{i+1}. Note that in the limiting case $v_{i+1} = v_{i-1}$ there are no wiggles, as $v_i = v_{i+1} = v_{i-1}$ follows immediately from (9).

Now assume two distinct cases (keeping in mind $h > 0$, $\epsilon > 0$),

- $v_{i+1} < v_{i-1}$: A wiggle is assured if either $v_i > v_{i-1}$

$$\begin{aligned} \Longleftrightarrow \frac{h}{8\varepsilon}(v_{i+1}^2 - v_{i-1}^2) &< \frac{v_{i-1} - v_{i+1}}{2} \\ \Longleftrightarrow \frac{h}{2\varepsilon} &> \frac{-2}{v_{i-1} + v_{i+1}} \\ \Longrightarrow \frac{h}{2\varepsilon} &> \frac{2}{|v_{i+1} + v_{i-1}|} \end{aligned}$$

or if $v_i < v_{i+1}$,

$$\begin{aligned} \Longleftrightarrow \frac{h}{8\varepsilon}(v_{i+1}^2 - v_{i-1}^2) &> \frac{v_{i+1} - v_{i-1}}{2} \\ \Longleftrightarrow \frac{h}{2\varepsilon} &> \frac{2}{v_{i-1} + v_{i+1}} \\ \Longrightarrow \frac{h}{2\varepsilon} &> \frac{2}{|v_{i+1} + v_{i-1}|} \end{aligned}$$

- $v_{i+1} > v_{i-1}$ we A wiggle is assured if either $v_i > v_{i+1}$,

$$\begin{aligned} \Longleftrightarrow \frac{h}{8\varepsilon}(v_{i+1}^2 - v_{i-1}^2) &> \frac{v_{i+1} - v_{i-1}}{2} \\ \Longleftrightarrow \frac{h}{2\varepsilon} &> \frac{2}{v_{i-1} + v_{i+1}} \\ \Longrightarrow \frac{h}{2\varepsilon} &> \frac{2}{|v_{i+1} + v_{i-1}|} \end{aligned}$$

or if $v_i < v_{i-1}$,

$$\begin{aligned} \Longleftrightarrow \frac{h}{8\varepsilon}(v_{i+1}^2 - v_{i-1}^2) &< \frac{v_{i-1} - v_{i+1}}{2} \\ \Longleftrightarrow \frac{h}{2\varepsilon} & \quad \frac{-2}{v_{i-1} + v_{i+1}} \\ \Longrightarrow \frac{h}{2\varepsilon} &> \frac{2}{|v_{i+1} + v_{i-1}|} \end{aligned}$$

Thus,

$$\frac{h}{2\varepsilon} > \frac{2}{|v_{i+1} + v_{i-1}|} \tag{10}$$

whenever $\frac{2}{|v_{i+1}+v_{i-1}|}$ is defined.

Considering (10) it is likely to observe wiggles whenever the right hand side of (10) is small, i.e. where v is still increasing and but has not yet "leveled off". In practice this is exactly where the wiggles appear (figure 2).

From (10) a sufficient condition for the absence of wiggles can be found, namely

$$h < 2\varepsilon \tag{11}$$

The presence of wiggles indicates the qualitative character of the true steady state solution has been lost in the discrete steady state solution. The use of such a solution in the Arnoldi algorithm means virtually anything can happen. For example, it can be shown that the eigenvalues of **A** are real if $h < 2\varepsilon$, but it can not be shown (and is in reality not true) if $h < 2\varepsilon$.

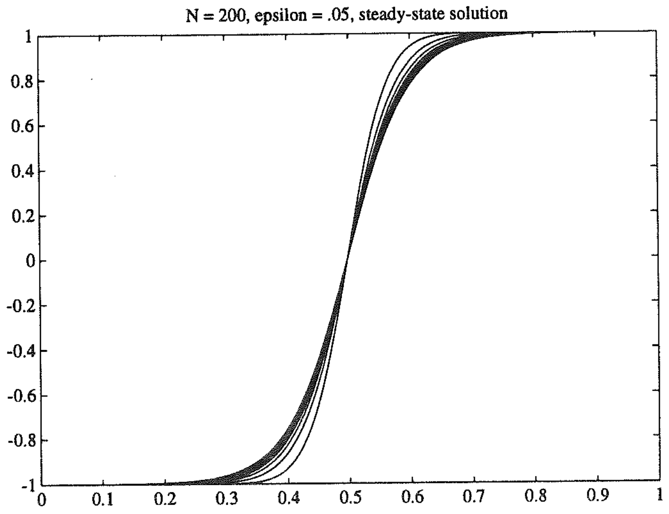

Figure 1: Convergence of quasi-steady state solution with $\varepsilon = 0.05, \quad N = 200$

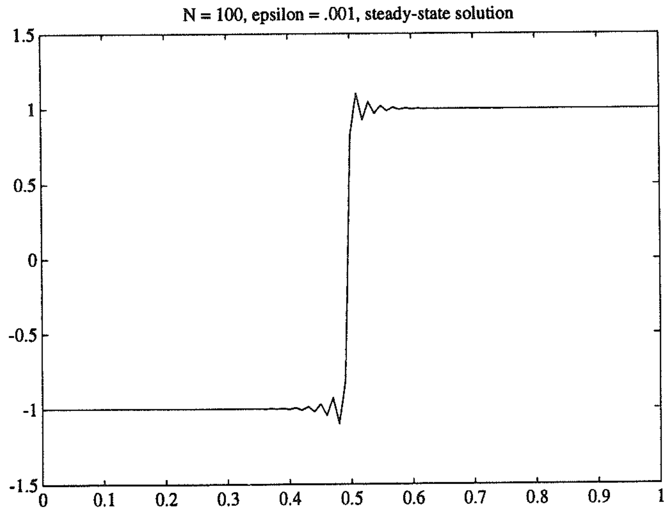

Figure 2: Numerically calculated steady state solution with $\varepsilon = 0.001, \quad N = 100$

3 The Eigenvalues of A

It has been shown that if $\frac{2\varepsilon}{h} > 1$ the numerically calculated values v_{st_i} increase monotonically from -1 to 1 (absence of "wiggles"). We assume monotonicity in the remainder of this section. In this case, $A_{(i,i-1)}$ and $A_{(i,i+1)}$ from (6) are positive. Since the mirrored elements in the first sub-diagonal have the same sign (+) it is easy to show that the matrix can be diagonally symmetrized and therefore all the eigenvalues of **A** are real. In addition, the matrix is *nearly* negative diagonally dominant:

$$\begin{aligned} A_{(i,i-1)} + A_{(i,i+1)} &= \frac{v_{i+1} - v_{i-1}}{2h} + \frac{2\varepsilon}{h^2} \\ &= \frac{v_{i+1} - v_{i-1}}{2h} - A_{(i,i)} \\ &\approx -A_{(i,i)} \end{aligned}$$

If it were actually negative diagonal dominant it would definitely have only purely negative real eigenvalues. In this case, $\mathbf{A} = \mathbf{B} + \mathbf{C}$ is the sum of a negative diagonally dominant matrix **B** and a (not necessarily small) perturbation matrix **C**.

The common eigenvalue inequalities, e.g. Gerschgorin disks [4], lead to bounds which are not tight enough. The need for a tight bound is reflected by the fact that as $h \to 0$, the largest eigenvalue also goes to zero.

Fortunately, a theorem about stability matrices can be used in spite of the proximity of λ_1 to zero:

Theorem 1 *If $A \in \Re^{n\times n}, a_{ij} \geq 0$ for all $i, i \neq j$ and there exist positive numbers $t_1, \ldots, t_n$ such that,*

$$\sum_{j=1}^{n} t_j a_{ij} \leq 0, (i = 1, \ldots, n)$$

*then **A** is semi-stable, i.e. $\forall i, Re(\lambda_i) \leq 0$.*

Proof: See [6], p. 159. ◇

Assuming that $\frac{2\varepsilon}{h} > 1$ it follows that in our case $a_{ij} \geq 0$ for all $i \neq j$. Furthermore, it is extremely useful to assume the symmetry of the true steady state solution for the numerically calculated quasi steady state, namely,

$$\begin{aligned} -v_0 &= v_N = 1 \\ -v_i &= v_{N-i} \\ v_{N/2} &= 0, \text{if } N \text{ even} \end{aligned} \tag{12}$$

Tests show that the numerically calculated pseudo-steady state does in fact have this quality *if N is odd.*

As a shorthand notation, we define

$$\delta_i = \frac{v_i h}{2\varepsilon}. \tag{13}$$

Since $\frac{2\varepsilon}{h} > 1$, δ_i increases monotonically and $v_1 < \delta_1$ and $v_n > \delta_n$. Furthermore, δ_i has the symmetry properties in (13).

The proof of the semi-stability of **A** proceeds now by the construction of a vector **t** which fulfills the condition (1). Assume N odd, therefore $n = N - 1 = 2m$ even. Let $t_m = t_{m+1} = k_m > 0$. Consider the rows m and $m+1$ of **A**:

$$\begin{aligned} t_{m-1}(1-\delta_{m-1}) - 2t_m + t_{m+1}(1+\delta_{m+1}) &\leq 0 \\ t_m(1-\delta_m) - 2t_{m+1} + t_{m+2}(1+\delta_{m+2}) &\leq 0 \end{aligned}$$

Adding the two inequalities and remembering that $\delta_{m+l+1} = -\delta_{m-l}$,

$$(t_{m-1} + t_{m+2})(1+\delta_{m+2}) + (\delta_{m+1} - 1)(t_m + t_{m+1}) \leq 0$$

Apparently we can force equality by letting,

$$t_{m-1} + t_{m+2} = 2k_m \frac{1-\delta_{m+1}}{1+\delta_{m+2}}$$

Due to the bounds on δ_i is is clear that the above is positive. Furthermore, without loss of generality, we can assume that $t_{m-1} = t_{m+2} = k_{m-1} (> 0)$.

By recursively continuing "from the center outward" we deduce the following relationship.

$$t_{m-l} + t_{m+l+1} = 2(t_{m-l+1} + t_{m+l}) \frac{1-\delta_{m+l}}{1+\delta_{m+l+1}}$$

or, assuming symmetry once more,

$$t_{m-l} = t_{m+l+1} = k_{m-l+1} \frac{1-\delta_{m+l}}{1+\delta_{m+l+1}} > 0 \tag{14}$$

By induction. Assume,

$$\begin{aligned} t_{m-l+1} = t_{m+l} &= k_{m-l+1} \\ t_{m-l+2} = t_{m+l+1} &= k_{m-l+1} \frac{1+\delta_{m+l}}{1-\delta_{m+l-1}} \end{aligned}$$

Considering the rows of **A**t $m - l + 1$ and $m + l$. Adding these two inequalities, we have

$$(t_{m-l} + t_{m+l+1})(1+\delta_{m+l+1}) - 4k_{m-l+1} + 2k_{m-l+1} \frac{1+\delta_{m+l}}{1-\delta_{m+l-1}}(1-\delta_{m+l-1}) \leq 0$$

Thus equality can be achieved if,

$$t_{m-l} = t_{m+l+1} = k_{m-l+1} \frac{1-\delta_{m+l}}{1+\delta_{m+l+1}}$$

We have just proved the following theorem:

Theorem 2 *If $t_m = t_{m+1} > 0$ and given the following recursive relationship (from the center, outward),*

$$t_{m-l} = t_{m+l+1} = t_{m-l+1} \frac{1-\delta_{m+l}}{1+\delta_{m+l+1}} \tag{15}$$

The **t** *so defined will satisfy the equation,*

$$\mathbf{A}t = \left\{ \begin{array}{c} \beta_1 \\ 0 \\ \vdots \\ 0 \\ \beta_n \end{array} \right\} \tag{16}$$

Proof: We need only to show that $\beta_1 \leq 0$ and $\beta_n \leq 0$. Consider the first and last row of (16):

$$\begin{aligned} -2t_1 + t_2(1+\delta_2) &= \beta_1 \\ (1-\delta_{n-1})t_{n-1} - 2t_n &= \beta_n \end{aligned}$$

But $t_1 = t_2 \frac{1+\delta_2}{1-\delta_1}$, therefore

$$\underbrace{(\frac{-2}{1-\delta_1} + 1)}_{<0} \underbrace{(1+\delta_2)}_{>0} \underbrace{t_2}_{>0} = \beta_1 < 0 \tag{17}$$

It is easily shown that,

$$\beta_n = \beta_1 (< 0)$$

◇

Assuming the absence of wiggles and N odd, we have the following theorem:

Theorem 3 **A** *is semi-stable.*

Proof: By choosing **t** as in equation (15) we can satisfy the conditions of Theorem 1. ◇

The vector **t** can be generated in a similar manner if N is even. The proof can be extended to show the stability of **A** by using the following theorem:

Theorem 4 *If the inequalities in Theorem 1 are strict, then A is stable.*

Proof: The proof of theorem 1 in [6] can be revised appropriately. ◇

Now there is a sufficient basis to prove the desired result:

Theorem 5 *In the absence of wiggles, the eigenvalues of* **A** *are in the left-half plane.*

Proof: Since β_1 is strictly less than zero, we can find a revised recursive relation which gives us a right side which is slightly less than zero.

Refining the recursive relationship (15) as

$$\begin{aligned} t_{m-l} = t_{m+l+1} &= (1-\epsilon)t_{m-l+1} \frac{1-\delta_{m+l}}{1+\delta_{m+l+1}} \\ 1 \gg \epsilon &> 0 \end{aligned}$$

This still insures that the t_i are positive. The right hand side of equation (16) is purely negative, including β_1 $(= \beta_n)$ if ϵ is chosen small enough:

$$\left(\frac{-2(1-\epsilon)}{1-\delta_1} + 1 \right) \underbrace{(1+\delta_2)}_{>0} \underbrace{t_2}_{>0} = \beta_1$$

which will be less than zero if

$$\epsilon < \frac{1}{2} + \frac{\delta_1}{2} \quad \text{which is necessarily} > 0.$$

◇

Therefore, under the assumption that the discrete steady-state solution increases monotonically, it is possible to find a $\varepsilon > 0$ such that $\beta_1 < 0$ and $\beta_n < 0$ in (16), and therefore that **A** is semi-stable. With these theoretical results in mind we proceed to the numerical calculation of the critical eigenvalues.

4 Numerical Calculation of the Eigenvalues of A

Since **A** is non-symmetric, large and sparse for numerous grid-points N, the Arnoldi method is chosen to find the eigenvalues nearest to the imaginary axis. Since this method does not require **A** explicitly, but rather only the matrix product $\mathbf{A}x$, we use instead of (5) the ***Fréchet*** derivative, e.g.

$$\mathbf{A}x = \left.\frac{\partial \mathbf{F}}{\partial \mathbf{v}}\right|_{\mathbf{v}_{\mathrm{st}}} \cdot \mathbf{x} \approx \frac{\mathbf{F}(\mathbf{v}_{\mathrm{st}} + \delta\mathbf{x}) - \mathbf{F}(\mathbf{v}_{\mathrm{st}} - \delta\mathbf{x})}{2\delta} \tag{18}$$

In order to increase the accuracy the fourth-order accurate scheme can be used,

$$\mathbf{A}x \approx \frac{-\mathbf{F}(\mathbf{v}_{\mathrm{st}} + 2\delta\mathbf{x}) + 8\mathbf{F}(\mathbf{v}_{\mathrm{st}} + \delta\mathbf{x}) - 8\mathbf{F}(\mathbf{v}_{\mathrm{st}} - \delta\mathbf{x}) + \mathbf{F}(\mathbf{v}_{\mathrm{st}} - 2\delta\mathbf{x})}{12\delta} \tag{19}$$

4.1 The Arnoldi Method

The Arnoldi Method can be considered to be a partial reduction of **A** to upper Hessenberg form. If we construct an orthonormal basis after k steps, $\mathbf{V}^{(k)} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k]$, for the Krylov subspace with initial vector $\mathbf{v}$, $K_k = \mathrm{span}[\mathbf{v}, \mathbf{A}v, \mathbf{A}^2\mathbf{v}, \ldots, \mathbf{A}^{k-1}\mathbf{v}]$, where $k < n$. We find the relationship for the residual vector $\mathbf{r}$ to be

$$\mathbf{AV} = (\mathbf{V}, \tilde{r}) \begin{pmatrix} \mathbf{H} \\ \beta\mathbf{e}_k^T \end{pmatrix} \quad \text{where } \beta = ||\mathbf{r}|| \quad \text{and } \tilde{\mathbf{r}} = \frac{1}{\beta}\mathbf{r} \tag{20}$$

H is a $k \times k$ Hessenberg matrix. Note that if $\beta = 0$, $\mathbf{V}^{(k)}$ spans an invariant subspace of **A**. With a good choice of the initial vector **v** and for not too large k, we hope that $||\mathbf{r}||$ will become small and that $\mathbf{V}^{(k)}$ will be a good approximation of an invariant subspace. The eigenvalues of $\mathbf{H}^{(k)}$ are then good approximations of those of **A**.

The algorithm in its simplest form can be found in [4]. One problem with the simple algorithm is the difficulty in getting the *desired* eigenvalues. Approximations for the eigenvalues with the largest absolute value tend to be returned first. See [7] for a discussion.

Methods for filtering out the desired eigenvalues are a topic of current research. Two different approaches for finding the eigenvalues with the smallest negative real part are treated here.

- Updating the starting vector **v** through the implicit application of polynomial filters as described in [8].
- A spectral transformation can be performed $\mathbf{D} = g(\mathbf{A})$, which maps the interesting portion of the spectrum (i.e. in our case near the imaginary axis) to the outer part of **D**'s spectrum.

The basic Arnoldi iteration is explained in [4]; we reproduce it here for completeness:

```
while $\beta \neq 0$
        $h_{j+1,j} := \beta$;  $\mathbf{v}_{j+1} := \mathbf{r}_j/\beta$;  $j = j + 1$
        $\mathbf{w} := \mathbf{A}v_j$;  $\mathbf{r} := \mathbf{w}$
        for $i := 1$ to $j$
                $h_{i,j} := \mathbf{v}_i^T\mathbf{w}$;  $\mathbf{r} := \mathbf{r} - h_{i,j}\mathbf{v}_i$
        end
        $\beta := ||\mathbf{r}||_2$
```

```
    if $j < n$
        $h_{j+1,j} := \beta$
    end
end
```

4.2 Numerical Algorithms to determine the Eigenvalues of A

Both of the modified Arnoldi methods share the same procedure up to the actual eigenvalue solver:

1. **Discretization:** Choose the discretization by choosing $h = 1/N$.
2. **Steady State:** Find a quasi-steady state solution by using a ODE solver on equation (2) until $\mathbf{v}_t$ is sufficiently small (i.e. its normed change in each step is less than some tolerance *tol*).
3. **Frechét derivative:** Generate an approximation of the Jacobian at the steady state solution found above using approximation (5). This implies deciding on the order of the approximation as well as assigning the value of δ in (18) or (19).

The evaluation of the Frechét derivative is equivalent to forming the matrix vector product $\mathbf{A}x$, which is required in either of the two following variations of the Arnoldi method.

K-Step Arnoldi Method with Polynomial Filters

Numerous problems arise in determining the eigenvalues closest to the imaginary axis using the Arnoldi method. The matrix $\mathbf{A}$ is large and sparse, but $\mathbf{H}$ will generally not be sparse. Since the desired eigenvalues are near to the origin, they will not be the first ones found. Therefore the plain Arnoldi method's space requirements are not known a priori and are presumably large, as is the expected amount of calculation.

Many proposals have been made to revise these deficiencies of the plain Arnoldi method. Saad in [7] has proposed an algorithm to explicitly restart the algorithm after k and using information from $\mathbf{Q}$ to construct a new initial vector.

Sorensen in [8] proposes a related idea. First k Arnoldi steps are performed. In a loop, p more Arnoldi steps are performed, then p "unwanted" eigenvalues (see below) are chosen and filtered out using the implicit shift algorithm (see [4] for details). The matrices $\mathbf{H}_{k+p}$, and $\mathbf{V}_{k+p}$ are partitioned into an unwanted parts $\hat{\mathbf{H}}_p, \hat{\mathbf{V}}_p$ and a desired part $\mathbf{H}_k^+, \mathbf{V}_k^+$. In short,

$$\mathbf{A}\mathbf{V}_k = (\mathbf{V}_k, \tilde{r}) \begin{pmatrix} \mathbf{H} \\ \beta_k \mathbf{e}_k^T \end{pmatrix} \underbrace{\longrightarrow}_{k \to k+p \to k} \mathbf{A}\mathbf{V}_k^+ = (\mathbf{V}_k^+, \tilde{r}^+) \begin{pmatrix} \mathbf{H}_k^+ \\ \beta_k^+ \mathbf{e}_k^T \end{pmatrix}$$

i.e. the new equation has exactly the same form as (20), meaning that the same process can be repeated again. This procedure has the result of implicitly restarting the algorithm with a new initial vector $\mathbf{v}^+$:

$$\mathbf{v} \longleftarrow \psi(\mathbf{A})\mathbf{v} \quad (= \mathbf{v}^+) \tag{21}$$

where the $\psi(\lambda) = (1/\tau) \prod_{j=1}^{p} (\lambda - \mu_j)$ is the filtering polynomial (τ is a normalization parameter). This "accordion" process of $k \longrightarrow k+p \longrightarrow k \longrightarrow k+p \cdots$ repeats until is $\beta_k^+ < tol$.

This process is not necessarily intuitive since it is not obvious that $\mathbf{H}_{k+p}$ and $\mathbf{V}_{k+p}$ can be partitioned into a good and bad part, nor is it readily apparent that the k "good" Ritz values

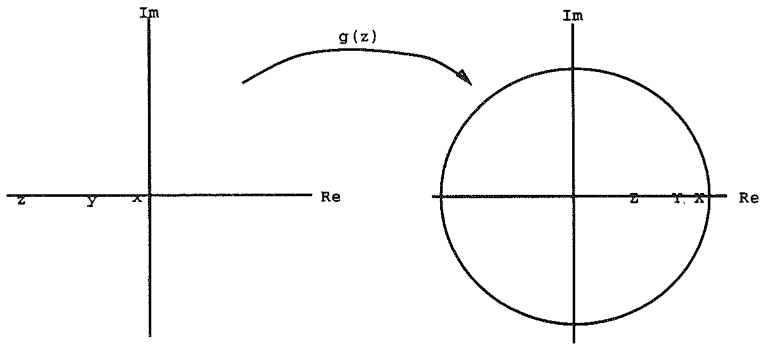

Figure 3: Spectral transformation $e^{\text{eig}(\mathbf{A})t}$

correspond to the actual eigenvalues of interest. The interested reader should refer to [8] for details.

Since k and p are constant and relatively small, The accordion process has the advantage that complete reorthogonalization of the Arnoldi vectors can be performed at a not unreasonable cost. Reorthogonalization avoids the intrusion of spurious eigenvalues.

The success of the filtering polynomial ψ strongly depends on the appropriate choice of μ_j. Sorensen [8] suggests the following:

- Exact shifts (i.e. μ_j are the p unwanted eigenvalues).
- ψ is constructed from a combination of Chebyshev polynomials.
- ψ is constructed to resemble a step function which is $= 0$ in the unwanted regions and 1 in the desired regions.

In our implementation, we use parameters chosen empirically depending on the problem size and nature (see discussion in section **Results**).

Arnoldi Method with Spectral Transformation

One possibility suggested in [2] for a spectral transformation is $C(t) = e^{\mathbf{A}t}$ (see figure 3). The eigenvalues closest to the imaginary axis will be mapped to points farthest from the origin.

For practicality, it is necessary to calculate a polynomial approximation of the exponential, e.g. the Gary [3], or second-order approximation:

$$\mathbf{C}(t) \approx \mathbf{D}(\Delta t, l) = (\mathbf{I} + \Delta t\mathbf{A} + \frac{1}{2}(\Delta t)^2\mathbf{A}^2 + \frac{1}{4}(\Delta t)^3\mathbf{A}^3)^l \tag{22}$$

where $t = l\Delta t$ and Δt is chosen such that the eigenvalues of $\mathbf{A}$ with largest modulus do not become a problem.

After determining the steady state solution, the algorithm is then the following,

1. **Spectral Transformation:** Perform the spectral transformation on the polynomial approximation of $e^{\mathbf{A}t}$ (as in (22)). This implies assigning values to Δt and l as well as the order of the approximation.

2. **Arnoldi Iteration:** Find the eigenvalues, hopefully with largest absolute value, by applying Arnoldi's method. These correspond to the eigenvalues of $\mathbf{A}$ with smallest negative real part. We can influence this process by changing the initial vector $\mathbf{v}$ and varying the number of iterations (i.e. the dimension k of the Hessenberg matrix $\mathbf{H}$).

3. **Reverse Transformation:** Transform the eigenvalues back with $\mathrm{eig}(\mathbf{A}) = \log \mathrm{eig}(\mathbf{C})/t$.

Given the values of ε and h it is necessary to find reasonable values for the parameters, δ, Δt, l, $\mathbf{v}$, k, as well as to decide on the tolerance *tol* and the orders of the Frechét and Polynomial approximations.

In order to choose Δt the eigenvalue of largest modulus must be considered. Keep in mind that it is much smaller than the eigenvalues of interest. Clearly this eigenvalue would map to a point very near the origin if we used the transformation $e^{\mathbf{A}t}$. In the polynomial approximation, however, this eigenvalue could map to one with *greater* absolute value than the eigenvalues of interest. Such an occurrence would defeat the purpose of the transformation. Consider the simplest polynomial approximation:

$$\begin{aligned} e^{\lambda \Delta t} &\approx 1 + \Delta t \lambda \\ \lambda_{\text{min}} &\ll 0 \end{aligned}$$

Clearly, if $\Delta t > \left|\frac{2}{\lambda_{\text{min}}}\right|$ the transform of eigenvalue λ_{min} will be outside the unit circle. The Arnoldi method will tend to find that eigenvalue instead of the ones of interest.

As a general rule, we can safely set

$$\Delta t = \left|\frac{1}{\lambda_{\text{min}}}\right|$$

in any of the of the polynomial approximations. The far left of the spectrum of $\mathbf{A}$ will be dominated by $\mathbf{B}$ in (7). It is known that,

$$0 > \mathrm{eig}(\mathbf{B}) > \frac{-4\varepsilon}{h^2}$$

and it is therefore safe to set,

$$\Delta t = \frac{h^2}{4\varepsilon} \tag{23}$$

A better approximation of λ_{min} can be found quickly by using a few iterations of the power method (see [4]).

The optimal value of t apparently varies inversely proportional to ε. To make this plausible, we offer the following argument. For good convergence of the eigenvalue estimates using the Arnoldi method, we would like,

$$\frac{|\mu_{i+1}|}{|\mu_i|} \leq 1 - \rho$$

where the $\mu_i = e^{\lambda_i(\mathbf{A})t}$ are the eigenvalues of the transformed system and ρ is not "too" small. If k eigenvalues are needed,

$$\frac{\mu_{k+1}}{\mu_k} = e^{-(\lambda_{k+1}-\lambda_k)t} \leq 1 - \rho$$

Although the distribution of the λ_i is not known a priori, the eigenvalues are real, negative and range from $-\frac{4\varepsilon}{h^2}$ to 0. Numerical experiments show that they do not bunch up as ε varies.

In fact, the basic distribution stays the same while their range increases with growing epsilon. Therefore it is safe to assume that,

$$(\lambda_{k+1} - \lambda_k) \sim \varepsilon$$

Therefore,

$$e^{-k\varepsilon t} \leq 1 - \rho$$

in other words, t should be inversely proportional to ε. An appropriate value for any particular t is found by manipulating l.

The value of l is a tradeoff between accuracy and efficiency: the larger l is, the larger the separation between the eigenvalues of the transformed system near the unit circle. On the other hand, the computational cost is increases with increasing l. It is conceivable that a smaller l will require more Arnoldi iterations before the required accuracy for the desired eigenvalue is found.

The initial vector $\mathbf{v}$ also has a direct effect on k, namely, a "good" initial vector allows for fewer Arnoldi iterations. A random vector could be used. It would be useful to have the initial vector close to an eigenvalue of $\mathbf{A}$. Such initial vector can be found be using consecutive iterates in the determination of the pseudo-steady state. Assume that v_n and v_{n-1} are close to each other and to the steady state v_{st}. Then,

$$\begin{aligned} (\mathbf{v}_n)_t &= F(\mathbf{v}_n) = F(\mathbf{v}_{n-1} + (\mathbf{v}_n - \mathbf{v}_{n-1})) \\ &\approx F(\mathbf{v}_{n-1}) + \left.\frac{\partial \mathbf{F}}{\partial \mathbf{v}}\right|_{\mathbf{v}_{n-1}} (\mathbf{v}_n - \mathbf{v}_{n-1}) \\ &\approx (\mathbf{v}_{n-1})_t + \mathbf{A}(\mathbf{v}_n - \mathbf{v}_{n-1}) \\ (\mathbf{v}_n - \mathbf{v}_{n-1})_t &\approx \mathbf{A}(\mathbf{v}_n - \mathbf{v}_{n-1}) \end{aligned}$$

Therefore, if the time steps are close enough together we are near enough to a steady state,

$$\frac{1}{\Delta t}(\mathbf{v}_n - \mathbf{v}_{n-1}) = \mathbf{A}(\mathbf{v}_n - \mathbf{v}_{n-1})$$

in other words, $\mathbf{v} = \mathbf{v}_n - \mathbf{v}_{n-1}$ is a fair estimation of an eigenvector and thus a better choice for the initial vector than a purely random guess.

The order of the polynomial approximation and the Frechét derivative as well as the tolerance *tol* are determined such that they do not effect the accuracy of the calculations. We used the second order Gary approximation and the fourth order Frechét derivative. The value of *tol* is chosen to be 10^{-6}.

A discussion of the proper value for δ can be found in [2]. The value is chosen to be 10^{-5}.

5 Numerical Results

It is possible to anticipate some of the results by analyzing the problem qualitatively.

- Consider the case $h \to 0$. The matrix $\mathbf{A}$ will become very nearly a symmetric matrix with constant diagonal and sub-diagonal elements with the proportion $1 : -2 : 1$. This is a well-known one-dimensional discrete Laplacian matrix with the eigenvalues,

$$\lambda_i = \frac{\varepsilon}{h^2}(-2 + 2\cos(i * h))$$

  for small i the $\cos i * h \approx 1 - \frac{i^2h^2}{2}$ and therefore

$$\lambda_i = \frac{\varepsilon i^2}{2}$$

  Therefore the largest eigenvalues will stay approximately constant for h very small.

- Consider the case $h \to 2\varepsilon, \varepsilon \ll 1$. It is clear that $v_1 \approx v_2 \approx 1$, i.e. the shock of the steady state solution is very "narrow." Taking (17) and insert (13) with $v_1 \approx v_2 \approx 1$ we find,

$$\beta_1 \approx \frac{-(2\varepsilon - h)^2}{(2\varepsilon + h)2\varepsilon}$$

  which goes to zero for $h \to 2\varepsilon$. This indicates that the corresponding $\mathbf{A}$ is close to singular. Therefore we expect at least one eigenvalue to approach zero.

In general, we search for the five eigenvalues with largest real part for cases where $h < 2\varepsilon$, where ε varies from the near shock value of 0.0125 to the very well-behaved value of 0.2. Other cases are explored in order to analyze the effect of different parameters.

5.1 Results with Exponential Transformation

We present the results for the method discussed in section 4.2. The largest eigenvalue (nearest to the origin) is calculated for $N = 100$ to $N = 800$. The size k of the Hessenberg matrix for these tests is held at 25 which seems to insure that the largest eigenvalue is always found. Δt took on the value such that the eigenvalue of largest modulus, estimated by a few iterations of the power method, is mapped by the exponential approximation to the value 0.8, avoiding its influence among the eigenvalues of interest.

An acceptable value of t (and therefore l) is determined heuristically — at a certain point an increase in l will not improve accuracy. A random initial vector is used in every case.

	Order N				$\Delta t * l$
ε	99	199	399	799	
0.025	$-1.3254D-07$	$-1.5638D-07$	$-1.6374D-07$	$-1.6369D-07$	0.032
0.05	$-1.7855D-03$	$-1.8069D-03$	$-1.8137D-03$	$-1.8154D-03$	0.016
0.1	$-1.3524D-01$	$-1.3536D-01$	$-1.3539D-01$	$-1.3539D-01$	0.008
0.2	$-9.5703D-01$	$-9.5703D-01$	$-9.5702D-01$	$-9.5702D-01$	0.004

In the first place we see that the largest eigenvalue λ_1 is virtually independent of the step size for all large N. In addition, note that as $\varepsilon \to \frac{h}{2}$, the largest eigenvalue seems to converge to zero, as expected.

The results in this table are quite impressive, but there are two major deficiencies in this method. First, backing off only slightly from the parameter Δt or trying to decrease t from the given optimum mentioned above, yields completely erratic behavior in the eigenvalues — this indicates the extreme sensitivity of the polynomial approximation of the exponential. Secondly, these results are computationally extremely expensive. For the final row in the table above, namely for $\varepsilon = 0.2$, we find,

$N =$	99	199	399	799
Matrix-vector multiplications	1362	1140	1095	1003
N-Vector operations	545611	1061422	2101836	3342848

Even though the Arnoldi method generally approximates the eigenvalues of largest modulus, this is not guaranteed. Experimentation tells us that this is sometimes not the case. In figure 4, the first three Ritz values are plotted against the size of the Hessenberg matrix k. As expected, the algorithm generally approximates eigenvalues closest to the imaginary axis, i.e. the transformed approximations are closest to the unit circle. However there are exceptions — some such eigenvalues are properly approximated in later iterations, as indicated by the "jumps" in the curves. These jumps indicate the point at which an intermediate eigenvalue starts to appear. Note that the all the Ritz values in these test are real or nearly real.

As mentioned in the previous section, the choice of the initial vector plays a noticeable, if not significant, role in the accuracy of the eigenvalues. For the case $N = 40$, $\varepsilon = 0.2$ and $\Delta t = 0.001$ and $k = 5$, we find the following five eigenvalue approximations for the two different initial vectors.

Initial vector	*Random*	$\mathbf{v}_n - \mathbf{v}_{n-1}$
λ_1	-0.9706	-0.9457
λ_2	-8.4504	-7.4739
λ_3	-18.2756	-17.2479
λ_4	-59.3304	-30.9016
λ_5	-179.4162	-63.6642

The cost of this calculation is approximately 6,500 matrix-vector multiplications. It turns out that only the first of the values in the middle column is an accurate approximation for an eigenvalue, while four of five in the right column are. To find similar accuracy for four eigenvalues with a random initial vector k must be increased to 8, involving about 10,000 matrix operations. A comparable improvement using $\mathbf{v} = \mathbf{v}_n - \mathbf{v}_{n-1}$ is observed for other values of N and ε.

5.2 Results with Filtering Polynomials

Along with the standard parameters N and ε, The tolerance for these results is set at 0.0001. Generally the subspace dimension r is kept below 1/10 of the size of the system (n). This differs from the previous method, where it is sufficient to keep the subspace dimension $k = 25$ constant.

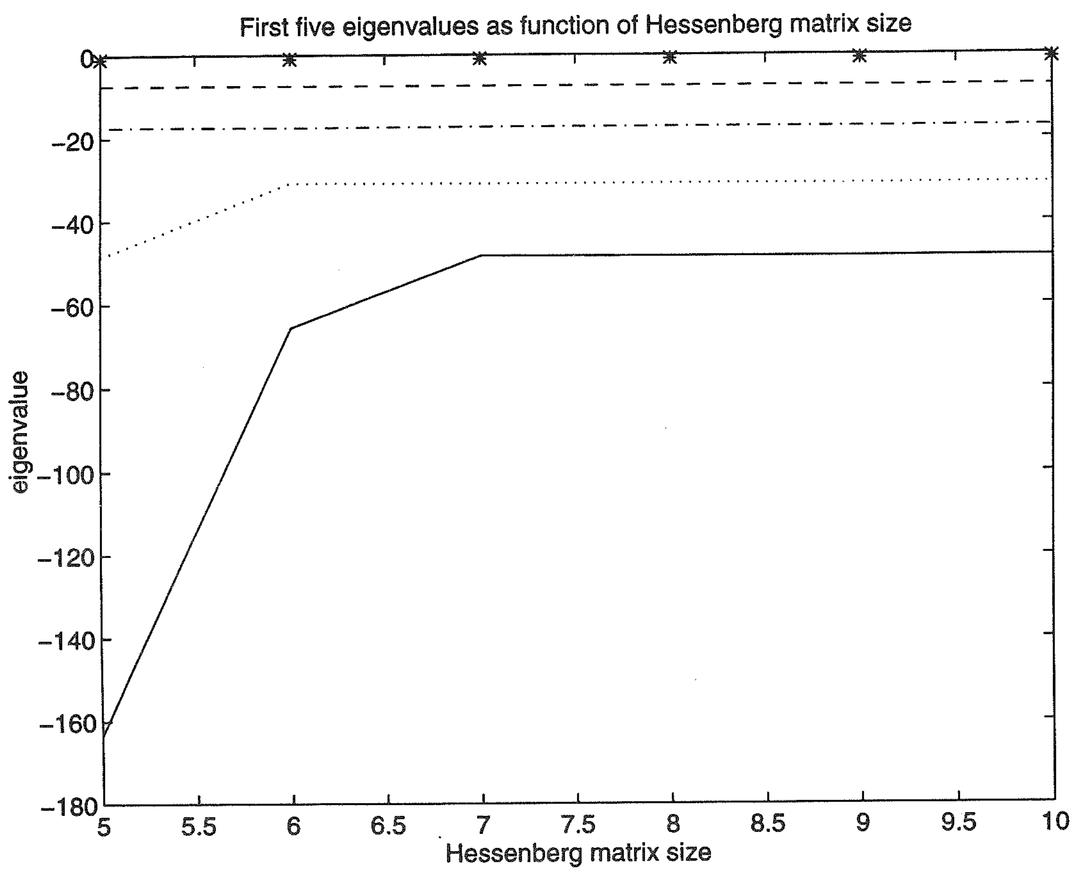

Figure 4: Five largest eigenvalues vs. size k of the Hessenberg matrix for $\varepsilon = 0.2$. Notice the eigenvalue "jumps" for small k.

	Order N			
ε	99	199	399	799
0.025	$-1.54955D-07$	$-1.78336D-07$	$-3.97836D-07$	$-4.94888D-07$
0.05	$-1.77973D-03$	$-1.80694D-03$	$-1.81372D-03$	$-1.81542D-03$
0.1	$-1.35242D-01$	$-1.35358D-01$	$-1.35387D-01$	$-1.35394D-01$
0.2	$-9.57063D-01$	$-9.57036D-01$	$-9.57030D-01$	$-9.57030D-01$

These values compare acceptably with the those from the exponential transformation method, although they seem to be somewhat less accurate. The very small positive values for the eigenvalues are still well within the given tolerance of 0.0001.

This algorithm is considerably more efficient than the previous one, even if one accounts for the additional overhead involved in polynomial filtering. For the calculation of the final row in the table ($\varepsilon = 0.2$), the following number of matrix vector multiplications are necessary,

$N =$	99	199	399	799
Matrix-vector multiplications	185	455	1245	3205
$r = k + p =$ Number of columns of V	25	35	45	55

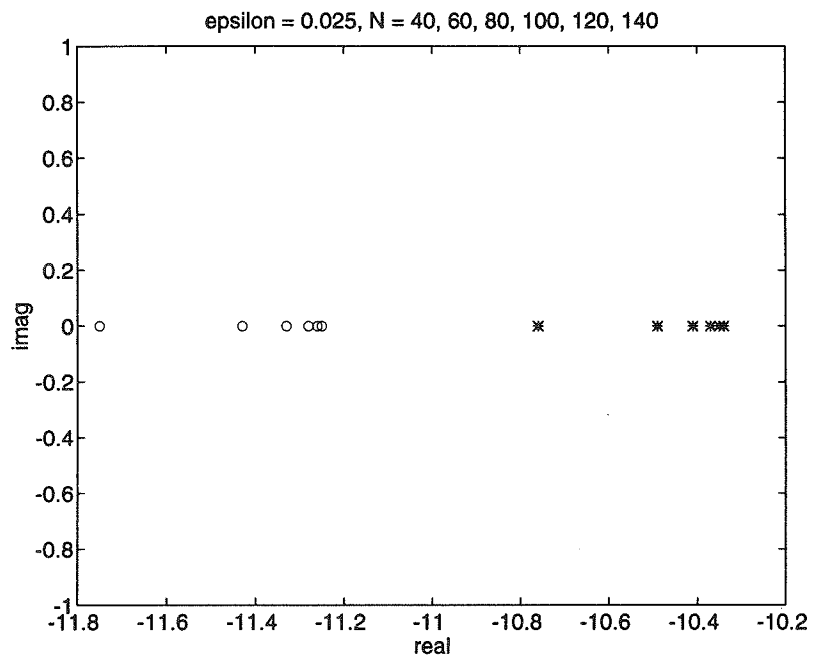

Figure 5: Second and third eigenvalues as function of problem size: $\varepsilon = 0.025$, λ_2 (symbol: $*$), λ_3 (symbol: o), depending on N.

The increase in number of matrix-vector multiplications for increasing N can be partially offset by using a larger $r = k + p$ (number of columns for V). Thus for $N = 799$, only 2735 multiplications are needed for $r = 70$.

Not only does the largest eigenvalue, but also neighboring eigenvalues converge well. Consider the second and third largest eigenvalues plotted in figure 5.

Even though all the cases treated should have purely real eigenvalues, it is possible, even likely, that for small ε numerical effects come into play which corrupt the eigenvalues. In figure 6, six eigenvalues are found for each of five different values for N with $\varepsilon = 0.0125$.

In each of these cases the eigenvalue with largest real part is incorrupted (it is approximately zero). The other eigenvalues appear as complex conjugate pairs until $N = 120$ (using $r = 20$) where the five largest eigenvalues are again entirely real.

One can see that this effect arises from numerical problems which can be avoided by increasing the size of the subspace (r) to an unrealistically large size. For $N = 80$, $r = 60$ and $N = 100$, $r = 55$ all the first five eigenvalues have "settled down" and are found to be real.

Even though a pseudo steady state solution with "wiggles" does not properly represent the true steady state, we found the largest eigenvalues for the case in figure 2. Recalling our analytical evaluation of the problem, the stability proof relies on the absence of wiggles $h < 2\varepsilon$ as well as the proof that the eigenvalues are real.

For $\varepsilon = 0.001, N = 100$ we find that the first three eigenvalues,

$$4.221 \times 10^{-8} \qquad -18.86 + 71.94i \qquad -18.86 - 71.94i$$

Even more interesting is the case $\varepsilon = 0.0125, N = 16$: In this case, wiggles are present and

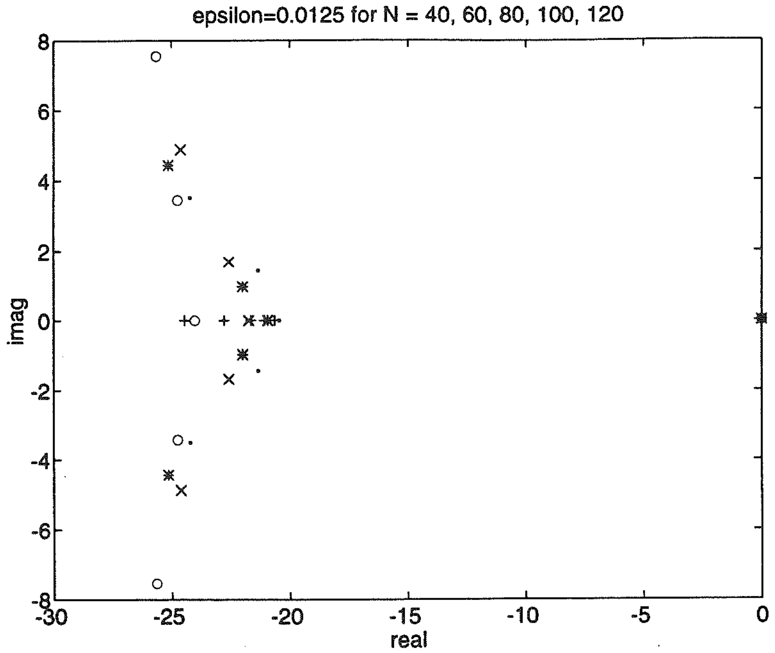

Figure 6: Six smallest eigenvalues as a function of problem size. Note the complex eigenvalues. $\varepsilon = 0.0125$, $N = 40$: o, 60: x, 80: *, 100: ., 120: +

there is an unmistakable *positive* eigenvalue.

$$\lambda_1 = +0.006591$$

There are, in addition, numerous complex eigenvalues (see Figure 7). This case serves as a example that a pseudo-steady state with wiggles has lost the qualitative nature of the problem.

5.3 Comparisons and Conclusions

It is difficult to compare the exponential transformation and the polynomial filtering algorithms, due to their very different nature.

The number of matrix-vector multiplications is favorable for the polynomial filtering algorithm for small problems. But since this operation can be computed at low cost for this problem due to the sparsity of the matrix, most of the expense is in other parts of both algorithms. While the number of floating point operations or vector operations would provide a more useful criteria, these data are not currently available from **ARPACK**.

A simple comparison can be made from wall-clock time. The polynomial filtering algorithm is consistently several times faster than the exponential transformation algorithm to obtain the same accuracy for the same input parameters ε and N, even if adjustments are made in the transformation method to incorporate a better-than-random initial guess and the subspace dimension k is decreased to the absolute minimum. This is still not a significant comparison, since neither of these algorithms have been optimized.

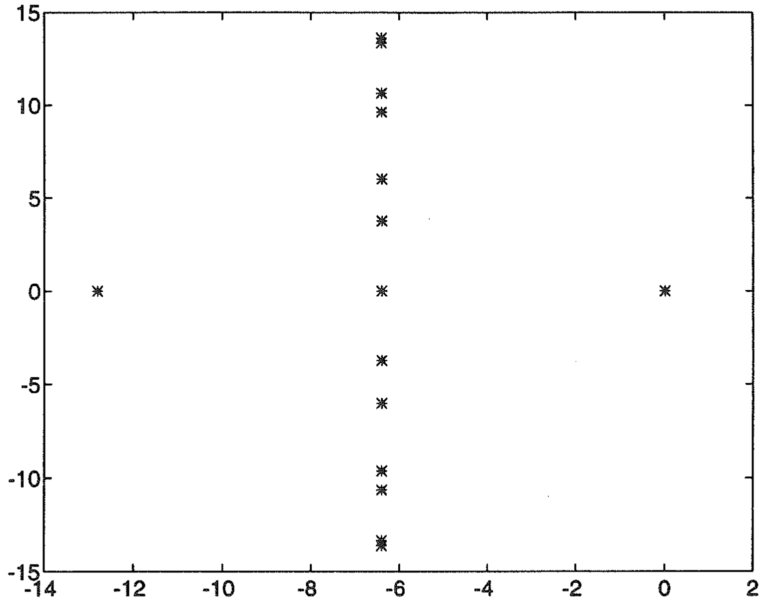

Figure 7: All eigenvalues of problem with "wiggles". $N = 16$, $\varepsilon = 0.0125$

On the other hand, the polynomial filtering algorithm used in **ARPACK** has clearly shown its usability, since it is available in an existing library and can be utilized with ease for this problem. It provides results which compare favorably with those of the exponential transformation algorithm, which requires considerably more design and development time.

References

[1] R. Lehoucq: *User's Guide for ARPACK on the Touchstone Delta.* Draft, **netlib** documentation for SCALAPACK library, 1994.

[2] L. E. Eriksson and A. Rizzi: *Computer-Aided Analysis of the Convergence to Steady State of Discrete Approximations to the Euler Equations.* Journal of Computational Physics **57**, 90 – 128 (1985).

[3] J. Gary: *On Certain Finite Difference Schemes for Hyperbolic Systems.* Math. Comp. , **18**, 1 – 18 (1964).

[4] G. Golub and C. Van Loan: *Matrix Computations.* Johns Hopkins University Press, Baltimore, 1989.

[5] P. Henrici: *Essentials of Numerical Analysis.* Wiley and Sons, New York, 1982.

[6] M. Marcus and H. Minc *A Survey of Matrix Theory and Matrix Inequalities.* Dover Publications, New York, 1964.

[7] Y. Saad *Chebyshev Acceleration Techniques for Solving Nonsymmetric Eigenvalue Problems*, Math. Comp., 42 (1984), pp. 567–588.

[8] D. Sorensen: *Implicit Application of Polynomial Filters in a k-step Arnoldi Method.* SIAM J. Matrix Analysis and Appl. **13** (1), 357 – 385 (1992).