Improving the Accuracy of Computed Eigenvalues
Using 32 and 64 bit floating point arithmetic

Jack Dongarra

Given a matrix $A$ and its eigenvalue, eigenvector pair $\lambda, x$ are by definition $Ax = \lambda x$.

A standing wave in a rope fixed at its boundaries can be seen as an example of an eigenvector, or more precisely, an eigenfunction of the transformation corresponding to the passage of time.
The Problem

♦ **Want to solve**

\[ Ax = \lambda x \]

♦ **But on a computer we make errors and don’t get \( \lambda, x \) exactly, but there exists \( \mu, y \) such that**

\[ A(x + y) = (\lambda + \mu)(x + y) \]
Given $\lambda$ & $x$ can we find $\mu$ & $y$

\[
A(x + y) = (\lambda + \mu)(x + y)
\]

Expanding things

\[
Ax + Ay = \lambda x + \lambda x + \mu x + \mu y
\]

Rearranging things

\[
(A - \lambda I)y - \mu x = \lambda x - Ax + \mu y
\]

\[
(A - \lambda I)y - \mu x = r + \mu y
\]

Term is square of error, will ignore

We have $(A - \lambda I)y - \mu x = r$

- At the moment we have $n+1$ unknowns $\mu, y$ and $n$ equations.
- Need one more equation.
- Eigenvectors can be normalized, we will choose $x_s=1$, $s$ is arbitrary.
- This imposes another constraint on the problem.
  - $n+1$ unknowns and $n+1$ equations
We Have \( (A - \lambda I)y - \mu x = r \)

- Rewrite the equation in matrix form with the constraint \( x_s = 1 \) or \( y_s = 0 \).

\[
\begin{pmatrix}
(A - \lambda I) & -x \\
e^T_s & 0
\end{pmatrix}
\begin{pmatrix}
y \\
\mu
\end{pmatrix} =
\begin{pmatrix}
r \\
0
\end{pmatrix}
\]

\[e^T_s y = 0\]

\[\mu y\] is a term in the error squared. We will ignore it.

We Have a Symmetric Eigenvalue Problem

- For a symmetric eigenvalue problem, the eigenvalues are real and the eigenvectors are orthogonal.
- The matrix, \( A \), can be reduced to a similar form, tridiagonal.
- The reduction is done by a sequence of orthogonal transformations, called \( Q \).
Let say we have computed the reduction to tridiagonal form

- We will do this in 32 bit arithmetic
- $O(n^3)$ ops
- $Q^T A Q = T$
At This Stage…

- We have the reduction to tridiagonal form
  - $Q^T AQ = T$
- And we compute the eigenvalues of $T$ in 32 bit floating point arithmetic.
- Now we want to compute a more accurate eigenvalue and an eigenvector

\[
\begin{pmatrix}
A - \lambda I & -x \\
\; e_s^T & 0
\end{pmatrix}
\begin{pmatrix}
y \\
\mu
\end{pmatrix} =
\begin{pmatrix}
r \\
0
\end{pmatrix}
\]

- Multiplying by $Q^T$ on both sides and using
  - $Q^T AQ = T$ and
  - $Q^T Q = QQ^T = I$

\[
\begin{pmatrix}
Q^T \\
1
\end{pmatrix}
\begin{pmatrix}
A - \lambda I & -x \\
\; e_s^T & 0
\end{pmatrix}
\begin{pmatrix}
Q \\
1
\end{pmatrix}
\begin{pmatrix}
y \\
\mu
\end{pmatrix} =
\begin{pmatrix}
Q^T \\
1
\end{pmatrix}
\begin{pmatrix}
r \\
0
\end{pmatrix}
\]

Identity
The matrix is a rank 2 modification of a tridiagonal matrix. 

Easy to solve

---

**Approach**

- Reduce the matrix to tridiagonal form in single precision.
- Compute the eigenvalues in single precision.
- Solve the tridiagonal system in an iterative step to improve the accuracy of the eigenvalue and compute the eigenvector.
  - Do this for each eigenvalue, eigenvector pair
- Process is equivalent to Newton's method
  - Quadratic convergence
- Requires 1.5 X the storage (one copy of the matrix in double precision and another in single precision)
Algorithm (For each eigenpair)

- **Reduce the matrix** $A$ **to tridiagonal form** $T$ **with a set of transformations** $Q$
- **Compute the eigenvalues of** $A$
  - $T = Q^T A Q$  \(\text{\% Reduce the matrix to tridiagonal form SINGLE } O(n^3)\)
  - $\Lambda = \text{eig}(T)$  \(\text{\% Fine the eigenvalues of } T \text{ SINGLE } O(n^3)\)
  - $r = \lambda x - Ax$  \(\text{\% Residual DOUBLE } O(n^2) \text{ with random } x\)
  - Form $B = (T - \lambda I, -Q^T x)$  \(\text{\% SINGLE } O(n^2)\)
    
    \[
    \begin{pmatrix}
    e^T_s Q & 0 \\
    
    \end{pmatrix}
    \]
  - $(L, U) = \text{LU}(B)$  \(\text{\% Factor the matrix } B \text{ SINGLE } O(n)\)
  - while ( \(|| r || \) not small enough ),  \(\text{\% stopping criteria}\)
    - $z = L \backslash (U \backslash r)$  \(\text{\%LU factorization on the residual SINGLE } O(n)\)
    - $x = x + Q z_{1:n}$  \(\text{\% new solution DOUBLE } O(n^2)\)
    - $\lambda = \lambda + z_{n+1}$  \(\text{\% new solution DOUBLE } O(1)\)
    - $r = Q^T (\lambda x - Ax)$  \(\text{\% new residual \text{\'DOUBLE } O(n^2)\}\)
  - end