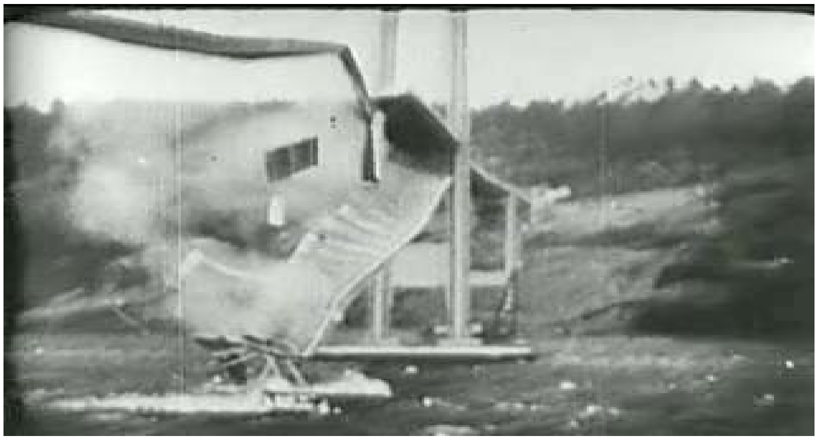
Improving the Accuracy of Computed Eigenvalues Using 32 and 64 bit floating point arithemetic

Jack Dongarra

Eigenvalues

- **Given a matrix A and its eigenvalue, eigenvector pair λ, 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.**

Singing Water Goblet



The Problem

- **Want to solve**

$$Ax = \lambda x$$

- **But on a computer we make errors and don't get λ, x exactly, but there exists μ, y such that**

$$A(x+y) = (\lambda + \mu)(x+y)$$

Given λ & x can we find μ & y

$$A(x+y) = (\lambda+\mu)(x+y)$$

Expanding things

$$Ax + Ay = \lambda x + \lambda x + \mu x + \mu y$$

Rearranging things

$$(A-\lambda I)y - \mu x = \underline{\lambda x - Ax} + \mu y$$

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

Term is square of error, will ignore

33 5

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

- At the moment we have **n+1** unknowns μ, 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

33 6

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.**

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

$$e_s=\begin{pmatrix} 0 \\ 0 \\ : \\ 1 \\ : \\ 0 \end{pmatrix}$$

← S element

$$e_s^T y=0$$

μy Is a term in the error squared
We will ignore it.

33 7

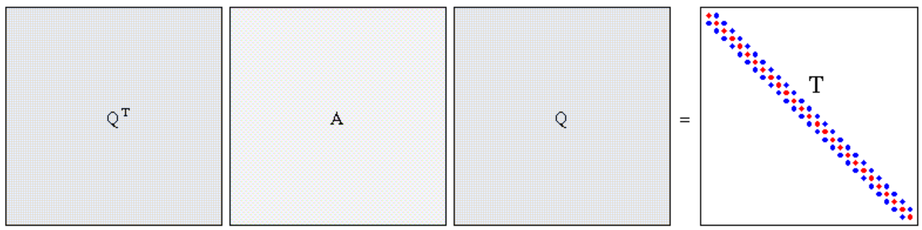
We Have a Symmetric Eigenvalue Problem

- **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.**

33 8

Reduction to Tridiagonal Form

$Q^T A Q = T$



$$\text{cost} : \approx \frac{4}{3} n^3 \text{ flops}$$

33 9

- Let say we have computed the reduction to tridiagonal form
  - We will do this in 32 bit arithmetic
  - $O(n^3)$ ops
- $Q^T A Q = T$

33 10

At This Stage…

- We have the reduction to tridiagonal form
  - $Q^TAQ = T$
- 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
  - $Q^TAQ = T$ and
  - $Q^TQ = QQ^T = I$

$$\begin{pmatrix} Q^T & \\ & 1 \end{pmatrix}\begin{pmatrix} A-\lambda I & -x \\ e_s^T & 0 \end{pmatrix}\underbrace{\begin{pmatrix} Q & \\ & 1 \end{pmatrix}\begin{pmatrix} Q^T & \\ & 1 \end{pmatrix}}_{\text{Identity}}\begin{pmatrix} y \\ \mu \end{pmatrix} = \begin{pmatrix} Q^T & \\ & 1 \end{pmatrix}\begin{pmatrix} r \\ 0 \end{pmatrix}$$

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

$$\begin{pmatrix} T-\lambda I & -Q^T x \\ e_s^T Q & 0 \end{pmatrix}\begin{pmatrix} Q^T y \\ \mu \end{pmatrix} = \begin{pmatrix} Q^T r \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} \alpha_1 & \beta_2 & & & & * \\ \beta_2 & \alpha_2 & \beta_3 & & & * \\ & \ddots & \ddots & \ddots & & \vdots \\ & & \beta_{n-1} & \alpha_{n-1} & \beta_n & * \\ & & & \beta_n & \alpha_n & * \\ * & * & * & * & * & 0 \end{pmatrix}$$

- 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 computer the eigenvector.
  - Do this for each eigenvalue, eigevector 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**
  - $T = Q^TAQ$ %Reduce the matrix to tridiagonal form SINGLE $O(n^3)$
  - $\Lambda = eig(T)$ % Fine the eigenvalues of T SINGLE $O(n^2)$
  - $r = \lambda x - Ax$ % Residual DOUBLE $O(n^2)$ with random x
  - Form $B = \begin{pmatrix} T - \lambda I, & -Q^Tx \\ e_s^TQ, & 0 \end{pmatrix}$ % SINGLE $O(n^2)$
  - (L,U)=LU(B) % Factor the matrix B SINGLE O(n)
  - while (|| r || not small enough), %stopping criteria
    - $z = L\backslash(U\backslash r)$ %LU factorization on the residual SINGLE O(n)
    - $x = x + Qz_{1:n}$ % new solution DOUBLE $O(n^2)$
    - $\lambda = \lambda + z_{n+1}$ % new solution DOUBLE O(1)
    - $r = Q^T(\lambda x - Ax)$ % new residual DOUBLE $O(n^2)$
  - end