

# The Eigenproblem and the Man in the Yellow Hat

**Jason Slemons**

University of Washington

Department of Applied Mathematics

29 November 2005

# Overview

- Eigenvalues of a general matrix
- The Symmetric Eigenproblem
- Details on Methods
  - QR iteration
  - Divide and Conquer
  - other things I think are neat.
- dqds
  - Symmetric Positive Definite case
  - Problems with the Symmetric Indefinite and non Symmetric cases

# Eigenvalues of a Matrix $A$

- State problem. Any ideas? exact solvers.

# Eigenvalues of a Matrix A

- State problem. Any ideas? exact solvers.
- Power Method:

$$u_{k+1} = Au_k \quad (1)$$

$$u_{k+1} = \frac{u_{k+1}}{\|u_{k+1}\|} \quad (2)$$

$$\lambda_{k+1} = u_{k+1}^T Au_{k+1} \quad (3)$$

$$k = k + 1 \quad (4)$$

What if we use  $A^{-1}$  instead of  $A$ ?

# Eigenvalues of a Matrix A

We should probably call in the inverse Power Method. or Jasons Method. either one.

- $(A - kI)^{-1}$  has evals  $\frac{1}{\lambda_1 - k}, \frac{1}{\lambda_2 - k}, \frac{1}{\lambda_3 - k}, \dots, \frac{1}{\lambda_n - k}$ . This gives the evaluate closest to k.
- Orthogonal iteration: start with  $Z_0$  and do the power method on it.  
 $AZ_i = Y_{i+1}$  then QR factorize  $Y_{i+1}$  to get  $Y_{i+1} = Z_{i+1}R_{i+1}$ .  
 $\text{span}\{Z_{i+1}\} = \text{span}\{A^{i+1}Z_0\}$
- Orthogonal iteration gives you the highest  $p$  evals if  $Z_0$  is  $n \times p$

## YAM- yet another method.

- QR iteration. Its okay.

$$A_i = Q_i R_i \quad (5)$$

$$A_{i+1} = R_i Q_i \quad (6)$$

$$i = i + 1 \quad (7)$$

- notice that  $A_{i+1} = R_i Q_i = Q_i^T (Q_i R_i) Q_i = Q_i^T (A_i) Q_i$ . This converges. Can add shifts to help convergence.
- $O(n^3)$  flops for one iteration? even if we just had one iteration per eigenvalue this is  $O(n^4)$ . reduce to Upper Hessenberg ( $\frac{10}{3}n^3 + O(n^2)$ ) and then one iteration of QR is only  $6n^2 + O(n)$  work.

# The Symmetric Eigenproblem

## Definition of Problem

- Different kinds(QR iteration, bisection, divide and conquer, dqds, holy grail, etc.)
- What they do
  - reduce to tridiagonal system
  - find e-vectors and values of the tridiagonal problem
  - back transform e-vectors

# ScaLAPACK Tridiagonalization

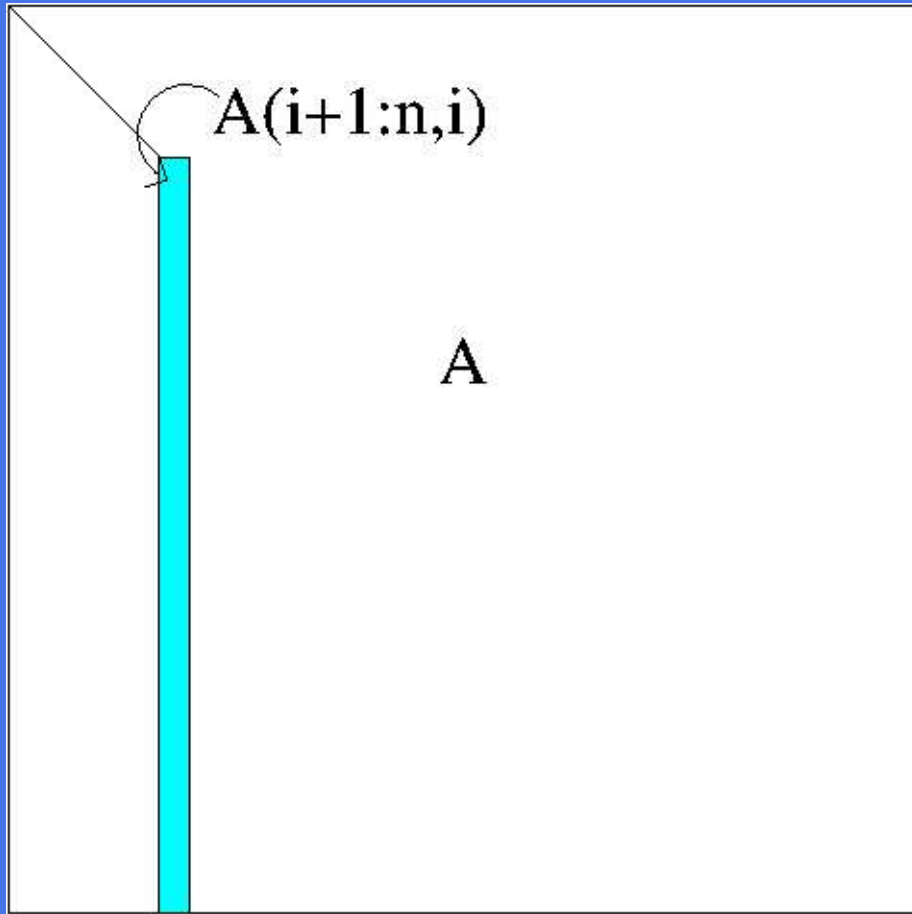
- Take the symmetric  $A$  and reduce it to a tridiagonal matrix  $HAH^T = T$  where  $H^T = H^{-1}$ , and orthogonal.
- ScaLAPACK uses a program called 'PDSYTRD'- Parallel Double precision SYmmetric TRiDiagonalization.
- Method: Householder Matrices. why?
- Same reduction to get a general matrix  $A$  to Upper Hessenberg.

# Serial Tridiagonalization

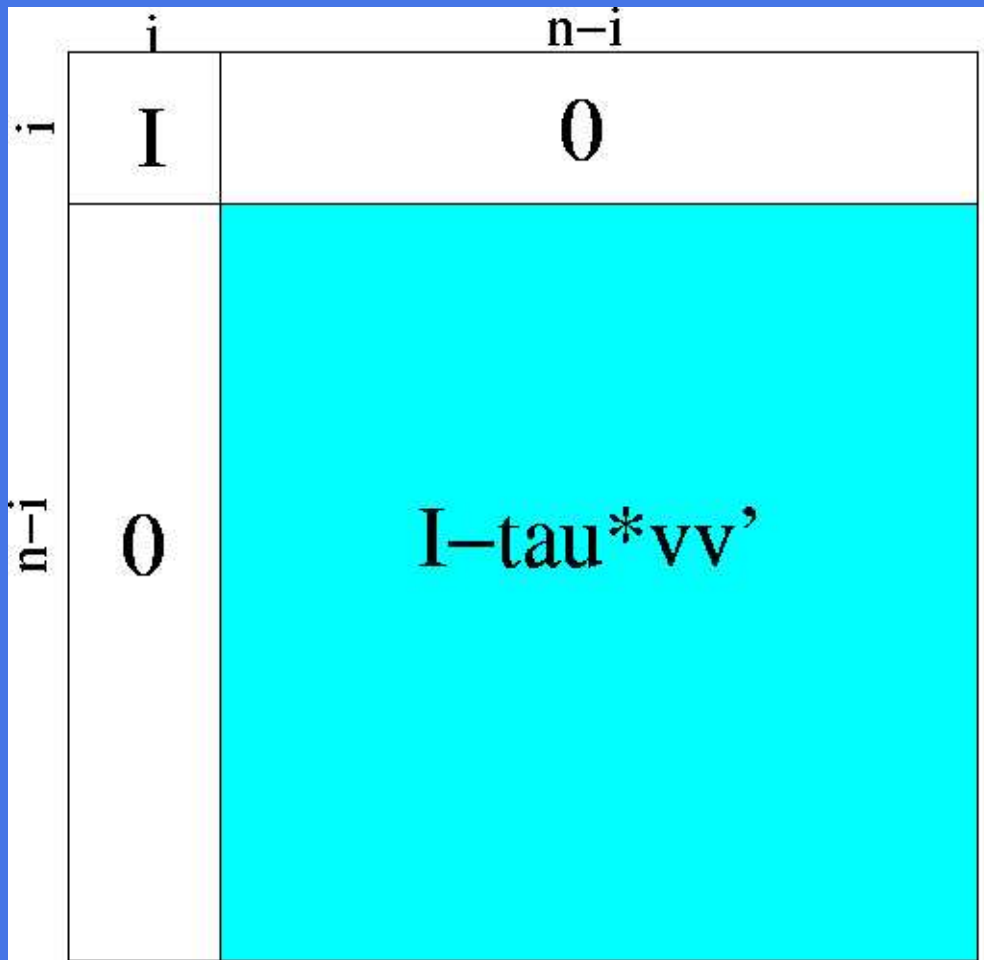
- Given a symmetric matrix  $A$
- A Householder matrix is  $H_1 = I - \tau vv'$  for a vector  $v$  and scalar  $\tau$  chosen so that  $H$  is orthonormal and so that  $H^t = H^{-1}$ .  $\tau = \frac{2}{vv'}$
- choosing  $v$  cleverly yeilds  $H_1AH_1^T$ , a matrix with a first row and column thats zero except for an entry on the diagonal and super/sub-diagonal.

# Serial Tridiagonalization

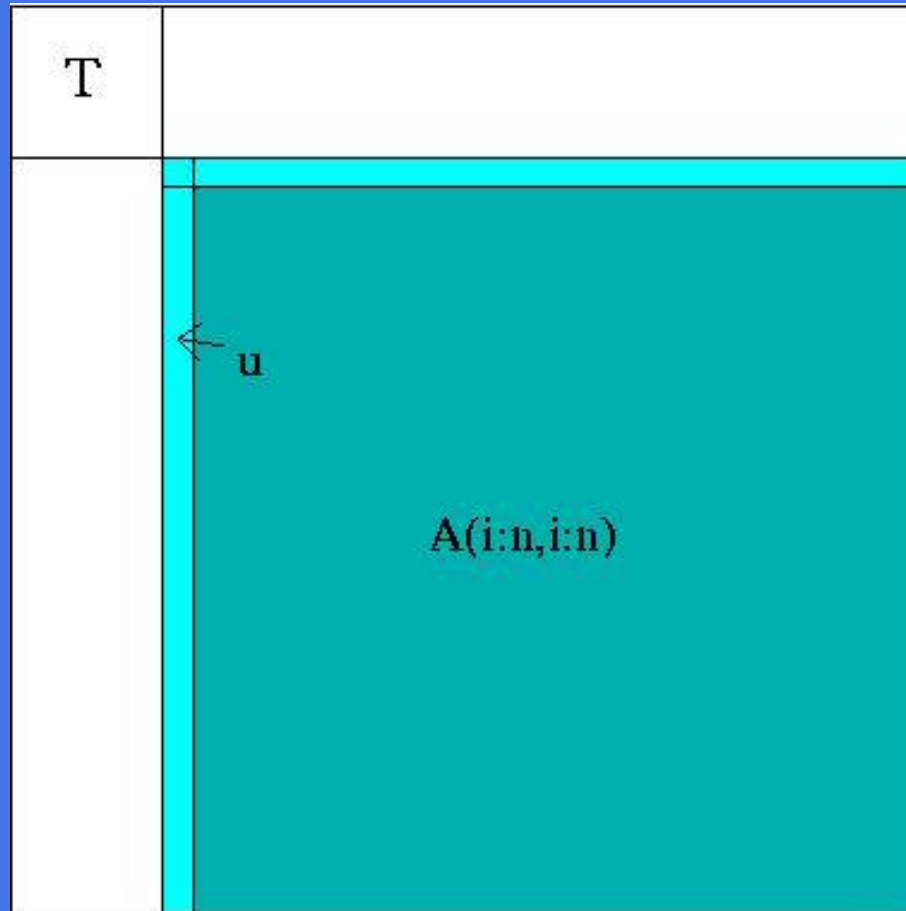
- Tridiagonalize lower  $n - 1$  by  $n - 1$  block of  $H_1AH_1$ .
- Eliminate the next row and column by another householder matrix;  $H_2$
- Repeat: across and down the matrix. Left with  $H_{n-2}\dots H_1A(H_{n-2}\dots H_1)^T = T$



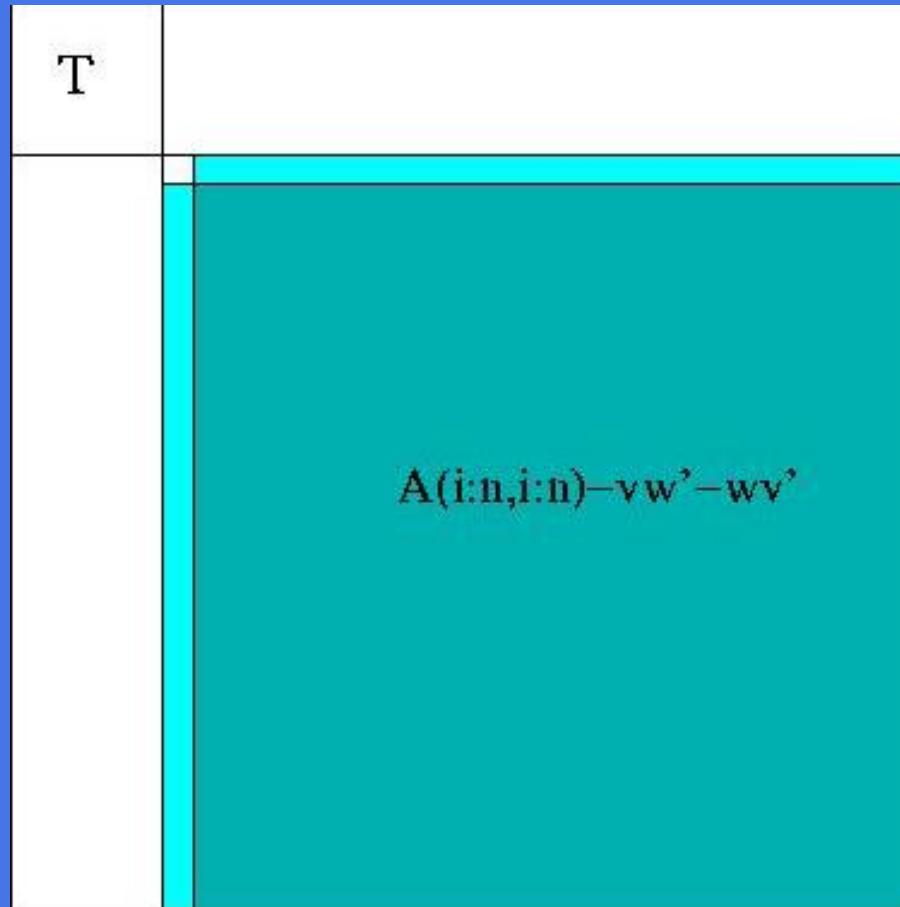
Set  $u = A_{i+1:n, i}$  then  $y = u \pm \|u\|e_1$  and  $v = y/y(1)$ .  $\tau = \frac{2}{vv'}$ .



From  $I - \tau v v'$  in the lower corner and  $I^i$  in the upper corner, form  $H$ .



$A$  is above. By choosing  $H$  we know  $H A H'$ 's  $i$ th column (labeled  $u$ ) will have only zeros after the sub-diagonal. What will the rest of the lower  $n - i$  corner of  $A$ ,  $A_{i:n,i:n}$ , be after conjugation? We need to find  $(H A H')_{i:n,i:n'}$ .



Form  $z = \tau A_{i:n, i:n} v$  and then  $w = z - \frac{\tau z' v}{2} v$ . The updated  $n - i$  block is of course, the mostly eliminated first column and the new  $A_{i:n, i:n}$  which is related by  $A_{i:n, i:n} = A_{i:n, i:n} - vw' - wv'$ .

# Serial Tridiagonalization

for  $i = 1 : n - 1$

1) Choose  $u = A(i + 1 : n, i)$

2)  $y = u \pm \|u\|e_1$

3)  $v = y/y(1)$

4) Calculate  $\tau = \frac{2}{vv'}$

IMPLICIT) Then form  $H$  from  $I - \tau vv'$  by adding an identity in the upper left corner of size  $i$ . find  $HAH'$

5)  $z = \tau * A_{i:n,i:n}v$  and  $w = z - \tau/2(z'v)v$ .  $A_{i:n,i:n}$  is the lower right block of  $A$  that's left.

6) Update the trailing matrix by  $A_{i:n,i:n} - vw' - wv'$ .

End when you have eliminated everything below the sub-diagonal. What's left is a tridiagonal matrix.

## Another Tridiagonalization

- The problem with tridiagonalization.  $O(n^3)$ , communication on parallel computers(my last talk)
- Use Bunch Kaufmann( $PAP^T = LDL^T$ ), Aasens( $PAP^T = LTL^T$ ), or Liu's Algorithm( $A = LDL^T$ ).
  - Speed like Cholesky, Accruate? Backward error of Cholesky is  $\|\delta A\|_\infty \leq 3n^2\epsilon\|A\|_\infty$ .
  - $D$  can have decent structure in the indefinite case. Else  $A = LL^T$ .
- Now look at  $L$  and bidiagonalize this using Jesse Barlowes algorithm ( A New Stable Bidiagonal Reduction Algorithm ) so now  $L = UBV^T$ .
  - $U$  has orthogonal columns(comes from  $Q$  in  $QR$ ).
  - $V$  is orthogonal(comes from  $H_n \dots H_1$  in a one sided tridiagonalization).  $B$  is bidiagonal.

## Another Tridiagonalization

- A indefinite:

$$A = LDL^T \quad (8)$$

$$A = (UB^T V^T)D(VBU^T) \quad (9)$$

$$V^T DV = I \quad \text{or} \quad V^T DV = D \quad (10)$$

$$A = UB^T BU^T \quad \text{or} \quad A = UB^T DBU^T \quad (11)$$

$$A = UTU^T \quad (12)$$

- A Positive Definite:

$$A = LL^T \quad (13)$$

$$QAQ^T = QLL^T Q^T \quad (14)$$

$$= (QL)(QL)^T \quad (15)$$

$$= T \quad (16)$$

# Summary of Tridiagonalization

- How to get  $V^T D V = D$  or  $I$ ? What kind of Householders can we use?
- $O(\frac{4}{3}n^3)$  work to reduce to tridiagonal form(2 sided) if we dont want the  $H_{n-2}\dots H_1$ ,  $O(\frac{8}{3}n^3)$  if we do. Tridiagonalization is worth it? yes.
- Which one though? Larger constant for one sided tridiagonalization. doh!
- 2 sided reduction on parallel computers; the devil you know.

# The Symmetric Tridiagonal Eigenproblem

- QR iteration with shifts takes  $O(n^2)$  w/o eigenvectors.  $O(n^3)$  with them. Fastest out there for  $n \leq 25$
- Divide and Conquer. Theoretically  $O(n^3)$  but in practice  $O(n^{2.3})$  with eigenvectors. best for  $n \geq 25$
- Holy Grail.  $O(n^2)$  for e values and e vectors. Pretty awesome.

# The Symmetric Tridiagonal Eigenproblem

- Bisection. start with an interval. Takes  $O(kn)$  where  $k$  is the number of eigenvalues. no eigenvectors.
- Inverse Iteration(Jasons Method). Works with Bisection. Eigenvectors arent great if the values are clustered. with shift of  $a_{nn}$  and  $u_0 = [0...0, 1]^T$  this is the same as QR iteration(with the Raleigh shift).
- dqds. Only for positive definite symmetric matrices. Its Awesome. differential quotient difference algorithm with shifts. lame name.

## dqds

- get symmetric tridiagonal via the above
- bidiagonal matrix  $B$  and find  $BB^T = T$ :

$i = 0$

repeat

Choose shift  $\tau_i^2$  smaller than the smallest eigenvalue of  $T_i$ .

Computer Cholesky factorization of  $T_i - \tau_i^2 I = B_i^T B_i$

$$T_{i+1} = B_i B_i^T + \tau_i^2 I$$

$$i = i + 1$$

until convergence

## dqds with Tridiagonal

- Start with Tridiagonal  $T$ , get the new  $\hat{T}$ , by rewriting the above as:

$$T - \tau^2 I = LU$$

$$UL - \tau^2 I = \hat{L}\hat{U} = \hat{T}$$

We can assume without loss that  $U_i$  has a diagonal of  $u_1, \dots, u_n$  and a superdiagonal of 1's.  $L_i$  has subdiagonal  $l_1, \dots, l_n$  and a diagonal of 1's.

$$L_i = \begin{pmatrix} 1 & & & & & \\ l_1 & \dots & & & & \\ & \dots & \dots & & & \\ & & l_i & 1 & & \\ & & & \dots & \dots & \\ & & & & l_n & 1 \end{pmatrix} \quad (17)$$

## dqds with Tridiagonal

$$U_i = \begin{pmatrix} u_1 & 1 & & & & \\ & u_2 & \cdots & & & \\ & & \cdots & 1 & & \\ & & & u_i & \cdots & \\ & & & & \cdots & 1 \\ & & & & & u_n \end{pmatrix} \quad (18)$$

$$L_i U_i = \begin{pmatrix} u_1 & 1 & & & & \\ l_1 u_1 & u_2 + l_1 & \cdots & & & \\ & \cdots & \cdots & 1 & & \\ & & l_i u_i & u_i + l_{i-1} & \cdots & \\ & & & \cdots & \cdots & 1 \\ & & & & l_n u_n & u_n + l_{n-1} \end{pmatrix} \quad (19)$$

## dqds

$UL - \tau^2 I = \hat{L}\hat{U}$  Written out element by element this is:

$$\hat{u}_1 = u_1 + l_1 - \tau^2 \quad (20)$$

$$\mathbf{for} \ i = 1, n - 1 \quad (21)$$

$$\hat{l}_i = \frac{l_i u_{i+1}}{\hat{u}_i} \quad (22)$$

$$u_{\hat{i}+1} = l_{i+1} + u_{i+1} - \tau^2 - \hat{l}_i \quad (23)$$

$$\mathbf{endfor} \quad (24)$$

## dqds

rewriting  $d_i = u_{i+1} - \hat{l}_i - \tau_i^2$  we get dqds:

$$d_1 = u_1 - \tau_1^2 \quad (25)$$

$$\mathbf{for} \ i = 1, n - 1 \quad (26)$$

$$\hat{u}_i = l_i + d_i \quad (27)$$

$$\hat{l}_i = \frac{l_i u_{i+1}}{\hat{u}_i} \quad (28)$$

$$d_i = d_i \left( \frac{u_{i+1}}{\hat{u}_i} \right) - \tau_i^2 \quad (29)$$

$$\mathbf{endfor} \quad (30)$$

## dqds

- d gets rid of subtractions.
- with correct shift and positive  $T$  all the d's are quantities are positive.
- amazing relative accuracy because basic operations are accurate. ( $6n * \epsilon$  normally its just  $O(K(T) * \epsilon)$ )
- who knows about machine epsilon?

## Indefinite case

- problems with shift strategy (cant shift to positive case)
- underflow/overflow checks
- settle for backward stable, fast. forget about accuracy?

## Non Symmetric case

- problems with factorization:  $LU$  factorization requires pivoting.
- shift strategy needs re-doing.
- stuck with QR iteration? lame.

# Future Work

- a banded version of dqds.
- formulate non symmetric dqds
- figure out why shifting doesnt destroy accuracy in Symmetric case
- indefinite shift strategy for dqds
- get PhD.

## Answers to Future Work

- a banded version of dqds....uh
- formulate non symmetric dqds: **most T factorize into LU so no problem there**
- figure out why shifting doesnt destroy accuracy in Symmetric case:

$$T_1 - \tau * I = L * U \quad (31)$$

$$T_2 = U * L + \tau * I \quad (32)$$

$$T_2 = L^{-1}(L * U + \tau * I)L \quad (33)$$

$$T_2 = L^{-1}(T_1)L \quad (34)$$

- indefinite shift strategy for dqds. **LU factorization requires diagonal dominates**