UNIVERSITY OF MINNESOTA TWIN CITIES

Numerical Methods in Electronic Structures Calculations

Yousef Saad

Department of Computer Science and Engineering

ITAMIT meeting – July 29th, 2005

Current & recent team members team: (CS side)

- Costas Bekas — Post-Doc
- Yunkai Zhou — Post-Doc
- Susanne Shontz — Post-Doc
- Shiv Gowda — Masters student [Now at NEC, Houston]
- Emmanuel Lorin de La Grandmaison — Post-doc [Now in Montreal]

What we do:

- We develop Numerical Algorithms
- We test them along with materials science researchers
- We optimize them
- We install them in materials code (s) – e.g., PARSEC
- .. a few years later ...

What we do:

- We develop Numerical Algorithms
- We test them along with materials science researchers
- We optimize them
- We install them in materials code (s) – e.g., PARSEC
- .. a few years later ...

We throw them away and repeat

What algorithms?

- Algorithms for solving eigenvalue problems
- Find alternatives [avoid eigenvectors, eigenvalues]
- Solve various related computational problems [TDDFT, computation of dielectric matrix, ...]

Electronic structures and Schrödinger's equation

- Determining matter's electronic structure can be a major challenge:

  Number of particules is large [a macroscopic amount contains $\approx 10^{23}$ electrons and nuclei] and the physical problem is intrinsically complex.

- Solution via the many-body Shrödinger equation:

$$H\Psi = E\Psi$$

- In original form the above equation is very complex

- Hamiltonian H is of the form :

$$H = -\sum_i \frac{\hbar^2 \nabla_i^2}{2M_i} - \sum_j \frac{\hbar^2 \nabla_j^2}{2m} + \frac{1}{2}\sum_{i,j} \frac{Z_i Z_j e^2}{|\vec{R}_i - \vec{R}_j|} - \sum_{i,j} \frac{Z_i e^2}{|\vec{R}_i - \vec{r}_j|} + \frac{1}{2}\sum_{i,j} \frac{e^2}{|\vec{r}_i - \vec{r}_j|}$$

- $\Psi = \Psi(r_1, r_2, \ldots, r_n, R_1, R_2, \ldots, R_N)$ depends on coordinates of all electrons/nuclei.

- Involves sums over all electrons / nuclei and their pairs

- Note $\nabla^2\Psi$ is Laplacean of Ψ = sum of second derivatives of Ψ in each direction. Represents kinetic energy.

A hypothetical calculation: (dont try this at home)

- 10 Atoms each having 14 electrons [Silicon]
- ... a total of 15*10= 150 particles
- ... Assume each coordinate will need 100 points for discretization..
- ... you will get

$$\#\,\text{Unknowns} = \underbrace{100}_{part.1} \times \underbrace{100}_{part.2} \times \cdots \times \underbrace{100}_{part.150} = 100^{150}$$

- Methods based on this basic formulation are limited to a few atoms.

Several approximations/ theories used

Problem can be viewed from the angle of optimization:

Find Ψ Minimize energy

$$\frac{< \Psi \mid H \mid \Psi >}{< \Psi \mid \Psi >}$$

.. or from the angle of eigenvalue problems:

Find eigenfunction Ψ associated with smallest eigenvalue of H

- **Methods have been developed on both camps**
- **Our camp: eigenvalue problems**

A brief history of diagonalization methods

- First: what is an eigenvalue problem?

  Given a matrix A, find a scalar λ and a nonzero vector x such that

  $$Ax = \lambda x$$

Quiz: what are the main uses of eigenvalues/ eigenvectors

Eigenvalue Problems. Their origins

- **Structural Engineering [$Ku = \lambda Mu$]**
- **Stability analysis [e.g., electrical networks, mechanical system,..]**
- **Bifurcation analysis [e.g., in fluid flow]**
- **Electronic structure calculations [Shrödinger equation..]**
- **Application of new era: page ranking on the world-wide web.**

Types of Problems:

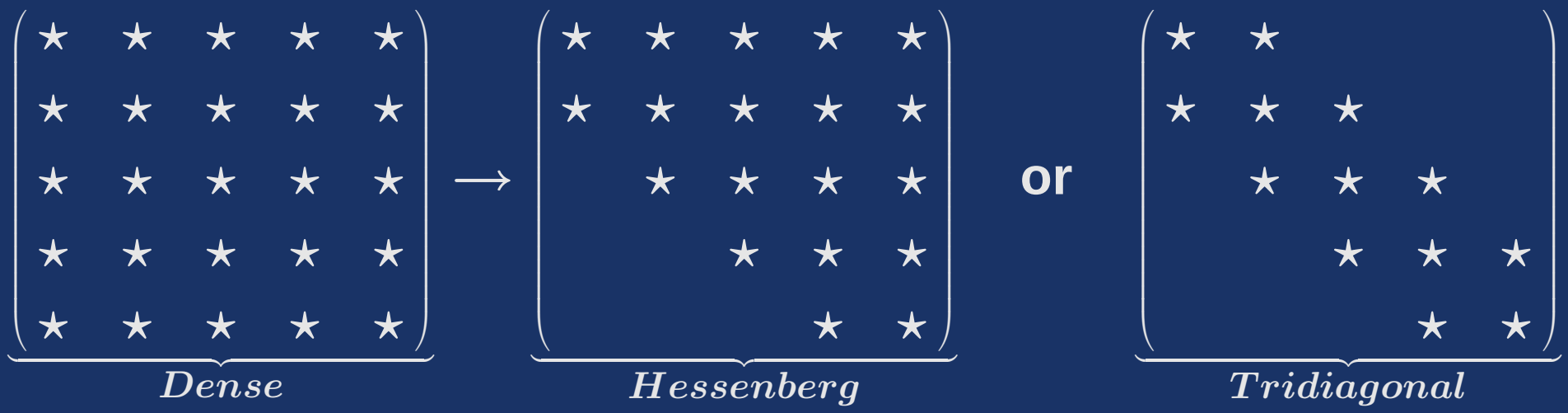
- Compute a few λ_i 's with smallest or largest real parts;
- Compute all λ_i's in a certain region of $\mathbb{C}$;
- Compute a few of the dominant eigenvalues;
- Compute all λ_i's.

Our problem: A is symmetric real – so its eigenvalues are real, and we want to compute the lowest k, where k represents the number of occupied states.

- k can be in the hundreds or thousands
- A can be very large up to a few Millions - currently.

Eigenvalues of dense matrices

- Early days: compute the characteristic polynomial.
- Works well “by hand” for dimension of up to 5 or so..
- Not viable for serious calculations
- Work in the 40’s and 50s concentrated on reducing the problem into an easy one: triadiagonal / or Hessenberg form (nonsymmetric matrices).



- Characteristic polynomials easier to compute..

Breakthroughs: The LR and QR algorithms

- Discovered in 1959 (LR) and 1961 (QR).
- Complicated to understand theoretically
- QR was held “secret” for a few years given its importance.
- Both QR and LR are very economical for tridiagonal matrices.

A brief history of eigenvalues - cont.

- 1965: Major book in computing eigenvalues by J. Wilkinson: The “bible” on eigenvalue problems
- 1971: volume by Wilkinson and Reinsch “handbook for automatic computations”, Linear algebra. Published 1st programs – in Algol... a now defunct language
- By 1975: EISPACK project. Translate Wilkinson & Reinsch volume in FORTRAN IV. Tremendous impact
- Later came LINPACK [linear equations]
- and much later came LAPACK = combined the two..

Current state of the art for dense computations:

- Consider the symmetric case only. Then
- Reduce A to tridiagonal form
- Use QR-alg. on tridiag matrix.
- Order n^3 calculation –
- software: LAPACK

Quiz: Suppose cost is $14n^3$ and $n = 10^6$, and you have a very powerful machine at home which can store the matrix and which delivers an operation every nanosecond (1 Gflop machine). How long would it take to compute the eigenvalues of A?

Quiz: Suppose cost is $14n^3$ and $n = 10^6$, and you have a very powerful machine at home which can store the matrix and which delivers an operation every nanosecond (1 Gflop machine). How long would it take to compute the eigenvalues of A?

Answer:

444 years!

Effective methods exploit sparsity

- Most methods exploit the fact that a product of a sparse matrix by a vector is very inexpensive [Order n]
- Suppose for the sake of argument that we now have an algorithm that takes exactly n^2 operations to compute one eigenvector.
- Same question as before for one eigenvector

Quiz: Suppose cost is n^2 and $n = 10^6$, and you have a very powerful machine at home which can store the matrix and which delivers an operation every nanosecond (1 Gflop machine). How long would it take to compute one eigenvector of A?

Quiz: Suppose cost is n^2 and $n = 10^6$, and you have a very powerful machine at home which can store the matrix and which delivers an operation every nanosecond (1 Gflop machine). How long would it take to compute one eigenvector of A?

Answer:

1000 sec = 16.667 mn!

Environments used throughout the project:

- Initially: Cray YMP [93]
- Cluster of SGI workstations
- CM5 [1994-1996]
- IBM SP2 [Using PVM]
- Cray T3D [Combining PVM + MPI] – around 1996-1997
- Cray T3E [using MPI] – 1997
- IBM SP with +256 nodes – 1998+
- IBM SP3 / SGI Altix currently