the
abdus salam
international centre for theoretical physics

SMR/1310 - 8

SPRING COLLEGE ON
NUMERICAL METHODS IN ELECTRONIC STRUCTURE THEORY

(7 - 25 May 2001)

"Numerical Linear Algebra - II" (NLA - II)

presented by:

D. ALFÈ

University College London
Department of Geological Sciences
and
Department of Physics and Astronomy
London
United Kingdom

Numerical Linear Algebra II

Dario Alfè

May 15, 2001

1 Linear systems, iterative methods

Consider the $N \times N$ linear system

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{b} \tag{1}$$

And assume for simplicity that the matrix $\mathbf{A}$ is symmetric and positive definite, i.e. $\mathbf{A} = \mathbf{A}^{\mathbf{T}}$ and $\mathbf{v} \cdot \mathbf{A} \cdot \mathbf{v} > 0 \ \forall \ \mathbf{v}$.

Consider now the function

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x} \cdot \mathbf{A} \cdot \mathbf{x} - \mathbf{b} \cdot \mathbf{x} \tag{2}$$

This function is minimised when its gradient is equal to zero

$$\nabla f(\mathbf{x}) = \mathbf{A} \cdot \mathbf{x} - \mathbf{b} = 0 \tag{3}$$

which is equivalent to (1). Notice that we don't need to know the matrix $\mathbf{A}$, but only its product by the vector $\mathbf{x}$. This is to an important advantage of this method over factorization methods in the case of large sparse matrices. To find the minimum of the function f we can use the method of the *conjugate gradient*, which is very efficient and allows one to find the minimum of f in at most N steps, where N is the dimensionality of the system (the size of the vector $\mathbf{x}$). Before we discuss the conjugate gradient algorithm it is instructive to consider first a different minimisation method, the *steepest descent.*

1.1 Steepest descent

The idea of the *steepest descent* (SD) method is to look for the minimum of the function starting from an arbitrary point and performing a series of minimisation steps along the directions of the SD of the function from that point. Suppose that we start at $\mathbf{x_0}$. The SD direction at $\mathbf{x_0}$ is given by the negative of the gradient of the function,

$$\mathbf{g} = -\nabla f(\mathbf{x_0}) \tag{4}$$

and one can find the minimum of f along the direction $\mathbf{g}$. This can be found with a line minimisation, i.e. sampling the function $f(\mathbf{x_0} + \alpha\mathbf{g})$ and looking for the minimum. One can also use a different approach: since at the minimum the gradient of the function is orthogonal to the searching direction one can look for the point where

$$\mathbf{g} \cdot \nabla f(\mathbf{x_0} + \alpha\mathbf{g}) = 0, \tag{5}$$

or

$$\mathbf{g} \cdot (A \cdot (\mathbf{x_0} + \alpha\mathbf{g}) - \mathbf{b}) = 0 \tag{6}$$

Notice that this minimisation *is not* a global minimisation of f, since it only minimises f along the direction $\mathbf{g}$. Although you go downwards every step, there is no guarantee that this procedure will bring you to the absolute minimum in a finite number of steps. Moreover, in some cases it can be very inefficient. Suppose you have a very narrow valley, if your starting search direction is neither parallel nor orthogonal to the main axis of the valley you end up doing a very large number of small steps before you can minimise the function. The reason is that any new searching direction is orthogonal to the previous one, so you will end up doing a large number of small steps across the valley. An other way of looking at this is that every new steps spoils the minimisation of the previous one, because the gradient of the function does not stay orthogonal to the gradient in the previous step, which means that the new minimum is no longer a minimum for the previous searching direction. This prompts the idea of the *conjugate method* that we discuss in the next section.

1.2 Conjugate gradient

In the *conjugate gradient* (CG) method the searching directions are arranged so as to avoid the problem of the SD method. As in the SD method one proceeds in a sequence of steps along straight lines, but the difference now is that the new searching direction is constructed so that the minimisations in the previous steps are not spoiled. Let's see how.

The first CG step is an SD step, there are no previous directions. The next searching direction is chosen so as to keep the gradient of the function orthogonal to the previous direction. So at the new minimum the gradient of the function is orthogonal *not only* to the actual searching direction *but also* to the previous searching direction, which means that the function now is minimised with respect to *both* the previous and the actual direction. If the dimensionality of the space is N one has to minimise the function with respect to N independent directions. Since every CG step minimises the function with respect to one dimension without spoiling the minimisation with respect to all the other dimensions, the maximum number of steps in the CG method is N. How do we construct the searching directions? The first one is minus the gradient of the function at the starting point: $\mathbf{h_0} = \mathbf{g_0} = -\nabla f(\mathbf{x_0})$. We then look for the minimum of f along this direction, which can be found with a line minimisation. If this is the global minimum of the function then we have $\nabla f(\mathbf{x_0} + \lambda_0 \mathbf{h_0}) = 0$ and we stop the search, otherwise we continue with the next step. Consider the gradient of f at $\mathbf{x_1} = \mathbf{x_0} + \lambda_0 \mathbf{h_0}$

$$\nabla f(\mathbf{x_1}) = \mathbf{A} \cdot \mathbf{x_1} - \mathbf{b} \tag{7}$$

We are looking for a direction $\mathbf{h_1}$ such that

$$0 = \nabla f(\mathbf{x_1} + \lambda \mathbf{h_1}) \cdot \mathbf{h_0} = (\mathbf{A} \cdot (\mathbf{x_1} + \lambda \mathbf{h_1}) - \mathbf{b}) \cdot \mathbf{h_0} \tag{8}$$

and since the previous step implies $(\mathbf{A} \cdot \mathbf{x_1} - \mathbf{b}) \cdot \mathbf{h_0} = 0$ we have

$$\mathbf{h_1} \cdot \mathbf{A} \cdot \mathbf{h_0} = 0 \tag{9}$$

and this is the condition that the searching directions have to satisfy, in which case $\mathbf{h_0}$ and $\mathbf{h_1}$ are called *conjugate directions.* The next steps have to be performed using conjugate directions, so that we must have

$$\mathbf{h_i} \cdot \mathbf{A} \cdot \mathbf{h_j} = 0, \qquad i \neq j \tag{10}$$

The following is a procedure to generate these conjugate directions, for more details see E. Polak, *Computational methods in Optimization*, Academic press, New York and London, 1971.

Conjugate Gradient Algorithm

0 Choose a starting vector $\mathbf{x_0}$. Set $\mathbf{g_0} = -\nabla f(\mathbf{x_0})$. If $\mathbf{g_0} = \mathbf{0}$ stop, else goto Step 1.

1 Set $i = 0$ and set the starting searching direction $\mathbf{h_0} = \mathbf{g_0}$.

2 Compute λ_i such that

$$f(\mathbf{x_i} + \lambda_i \mathbf{h_i}) = \min\{f(\mathbf{x_i} + \lambda \mathbf{h_i}) | \lambda \geq 0\} \tag{11}$$

and set $\mathbf{x_{i+1}} = \mathbf{x_i} + \lambda_i \mathbf{h_i}$.

3 Compute $\nabla f(\mathbf{x_{i+1}})$

4 If $\nabla f(\mathbf{x_{i+1}}) = 0$ stop; else set

$$\begin{aligned} \mathbf{g_{i+1}} &= -\nabla f(\mathbf{x_{i+1}}) \\ \mathbf{h_{i+1}} &= \mathbf{g_{i+1}} + \gamma_i \mathbf{h_i}, \qquad \text{with } \gamma_i = \frac{\mathbf{g_{i+1}} \cdot \mathbf{g_{i+1}}}{\mathbf{g_i} \cdot \mathbf{g_i}} \end{aligned} \tag{12}$$

set $i = i + 1$ and go to Step 2.

Notice that

$$\mathbf{g_{i+1}} = -\nabla f(\mathbf{x_{i+1}}) = -\mathbf{A} \cdot (\mathbf{x_i} + \lambda_i \mathbf{h_i}) + \mathbf{b} = \mathbf{g_i} - \lambda_i \mathbf{A} \cdot \mathbf{h_i} \tag{13}$$

and since (11) implies $\mathbf{g_{i+1}} \cdot \mathbf{h_i} = 0$ we obtain

$$\lambda_i = \frac{\mathbf{g_i} \cdot \mathbf{h_i}}{\mathbf{h_i} \cdot \mathbf{A} \cdot \mathbf{h_i}} \tag{14}$$

and step 4 can be restated as

$$\mathbf{g_{i+1}} = \mathbf{g_i} - \lambda_i \mathbf{A} \cdot \mathbf{h_i}; \qquad \mathbf{h_{i+1}} = \mathbf{g_{i+1}} + \gamma_i \mathbf{h_i} \tag{15}$$

We need to prove now that the searching directions constructed in (12) are indeed conjugate, i.e. they satisfy the relation (10). In order to do that let's define the following sequence of vectors:

$$\mathbf{g_{i+1}} = \mathbf{g_i} - \lambda_i \mathbf{A} \cdot \mathbf{h_i}; \qquad \mathbf{h_{i+1}} = \mathbf{g_{i+1}} + \gamma_i \mathbf{h_i}; \qquad \text{with } \mathbf{g_0} = \mathbf{h_0} \tag{16}$$

and λ_i, γ_i are chosen so that $\mathbf{g_{i+i}} \cdot \mathbf{g_i} = 0$ and $\mathbf{h_{i+i}} \cdot \mathbf{A} \cdot \mathbf{h_i} = 0$, i.e.

$$\lambda_i = \frac{\mathbf{g_i} \cdot \mathbf{g_i}}{\mathbf{g_i} \cdot \mathbf{A} \cdot \mathbf{h_i}}, \qquad \gamma_i = -\frac{\mathbf{g_{i+1}} \cdot \mathbf{A} \cdot \mathbf{h_i}}{\mathbf{h_i} \cdot \mathbf{A} \cdot \mathbf{h_i}} \tag{17}$$

This sequence is exactly the same as the one we want to use in the CG algorithm, but λ_i, γ_i are in a different form. By construction, $\mathbf{g_{i+1}} \cdot \mathbf{g_i} = \mathbf{h_{i+1}} \cdot \mathbf{A} \cdot \mathbf{h_i} = 0$. We shall prove that λ_i, γ_i are the same as those defined in the CG algorithm, and that there exist an integer $m < N$ such that for $i, j \leq m$

$$\mathbf{g_i} \cdot \mathbf{g_j} = \delta_{ij} ||\mathbf{g_i}||; \qquad \mathbf{h_i} \cdot \mathbf{A} \cdot \mathbf{h_j} = \delta_{ij} \mathbf{h_i} \cdot \mathbf{A} \cdot \mathbf{h_i} \tag{18}$$

and $\mathbf{g_i} = \mathbf{h_i} = \mathbf{0}\ \forall i > m$. This also ensures that the CG algorithm stops after at most N steps ($\mathbf{g_i} = \mathbf{0} \rightarrow \nabla f(\mathbf{x_i}) = 0 \rightarrow$ the function is at a minimum).

Proof. Suppose that we have generated a sequence $\mathbf{g_0}, \mathbf{g_1}, \ldots, \mathbf{g_m}$ and $\mathbf{h_0}, \mathbf{h_1}, \ldots, \mathbf{h_m}$ non zero vectors using the procedure defined in (16) and (17) and that $\mathbf{g_{m+1}} = 0$, then $\mathbf{h_{m+1}} = 0$ (see (16) and (17)). Now we prove that $\mathbf{h_{m+1}} = 0$ also implies $\mathbf{g_{m+1}} = 0$.

Form the scalar product $0 = \mathbf{g_{m+1}} \cdot \mathbf{h_{m+1}} = \mathbf{g_{m+1}} \cdot \mathbf{g_{m+1}} + \gamma_m \mathbf{g_{m+1}} \cdot \mathbf{h_m}$; we must prove that $\mathbf{g_{m+1}} \cdot \mathbf{h_m} = 0$. We do this by induction. We have $\mathbf{g_1} \cdot \mathbf{h_0} = 0$ ($\mathbf{g_0} = \mathbf{h_0}$). Now suppose that we have $\mathbf{g_j} \cdot \mathbf{h_{j-1}} = 0$ for all $j = 1, \ldots k$ for some integer $k \leq m$, we must prove that this implies $\mathbf{g_{k+1}} \cdot \mathbf{h_k} = 0$.

$$\begin{aligned} \mathbf{g_{k+1}} \cdot \mathbf{h_k} &= \mathbf{g_k} \cdot \mathbf{h_k} - \lambda_k \mathbf{h_k} \cdot \mathbf{A} \cdot \mathbf{h_k} = \\ \mathbf{g_k} \cdot (\mathbf{g_k} &+ \gamma_{k-1} \mathbf{h_{k-1}}) - \lambda_k \mathbf{h_k} \cdot \mathbf{A} \cdot \mathbf{h_k} = \\ &= \mathbf{g_k} \cdot \mathbf{g_k} - \lambda_k \mathbf{h_k} \cdot \mathbf{A} \cdot \mathbf{h_k} = 0 \end{aligned} \tag{19}$$

because $\mathbf{g_k} \cdot \mathbf{A} \cdot \mathbf{h_k} = (\mathbf{h_k} - \gamma_k \mathbf{h_{k-1}}) \cdot \mathbf{A} \cdot \mathbf{h_k} = \mathbf{h_k} \cdot \mathbf{A} \cdot \mathbf{h_k}$, and then

$$\lambda_k = \frac{\mathbf{g_k} \cdot \mathbf{g_k}}{\mathbf{h_k} \cdot \mathbf{A} \cdot \mathbf{h_k}} \tag{20}$$

So we have a sequence of nonzero vectors $\mathbf{g_0}, \mathbf{g_1}, \ldots, \mathbf{g_m}$, $\mathbf{h_0}, \mathbf{h_1}, \ldots, \mathbf{h_m}$ and $\mathbf{g_i} = \mathbf{h_i} = \mathbf{0}\ \forall i > m$. Now we prove that $\mathbf{g_i} \cdot \mathbf{g_j} = \mathbf{h_i} \cdot \mathbf{A} \cdot \mathbf{h_j} = 0$ if $i \neq j$, again using induction. This is trivially true for $i, j > m$, so we only need to prove it for $i, j \leq m$. By construction $\mathbf{g_1} \cdot \mathbf{g_0} = \mathbf{h_1} \cdot \mathbf{A} \cdot \mathbf{h_0} = 0$, now suppose that there exist an integer $k \leq m$ such that

$$\mathbf{g_i} \cdot \mathbf{g_j} = \mathbf{h_i} \cdot \mathbf{A} \cdot \mathbf{h_j} = 0, \qquad i \neq j, \qquad 0 \leq i, j \leq k \tag{21}$$

consider $i = 1, \ldots, k-1$,

$$\mathbf{g_{k+1}} \cdot \mathbf{g_i} = (\mathbf{g_k} - \lambda_k \mathbf{A} \cdot \mathbf{h_k}) \cdot \mathbf{g_i} = -\lambda_k (\mathbf{A} \cdot \mathbf{h_k}) \cdot \mathbf{g_i} = -\lambda_k (\mathbf{A} \cdot \mathbf{h_k}) \cdot (\mathbf{h_i} - \gamma_{i-1} \mathbf{h_{i-1}}) = 0 \tag{22}$$

$\mathbf{g_{k+1}} \cdot \mathbf{g_k} = 0$ by construction and

$$\mathbf{g_{k+1}} \cdot \mathbf{g_0} = (\mathbf{g_k} - \lambda_i \mathbf{A} \cdot \mathbf{h_i}) \cdot \mathbf{g_0} = -\lambda_i \mathbf{g_0} \cdot \mathbf{A} \cdot \mathbf{h_i} = 0 \qquad (\mathbf{g_0} = \mathbf{h_0}) \tag{23}$$

Similarly, $\mathbf{h_{k+1}} \cdot \mathbf{A} \cdot \mathbf{h_k} = 0$ by construction, and for $i = 0, \ldots, k-1$

$$\mathbf{h_{k+1}} \cdot \mathbf{A} \cdot \mathbf{h_i} = (\mathbf{g_{k+1}} + \gamma_k \mathbf{h_k}) \cdot (\mathbf{A} \cdot \mathbf{h_i}) = \mathbf{g_{k+1}} \cdot \mathbf{A} \cdot \mathbf{h_i} = \mathbf{g_{k+1}} \cdot \frac{\mathbf{g_i} - \mathbf{g_{i+1}}}{\lambda_i} = 0 \tag{24}$$

because $\lambda_i \neq 0$. We have a sequence of mutually orthogonal vectors $\mathbf{g_i}$ which can contain at most N elements, the dimensionality of the space, which means that $m < N$.

Now the last part of the proof: $\mathbf{g_i} \cdot \mathbf{g_i} = \mathbf{g_i} \cdot (\mathbf{h_i} - \gamma_{i-1} \mathbf{h_{i-1}}) = \mathbf{g_i} \cdot \mathbf{h_i}$, so that

$$\lambda_i = \frac{\mathbf{g_i} \cdot \mathbf{h_i}}{\mathbf{h_i} \cdot \mathbf{A} \cdot \mathbf{h_i}} \tag{25}$$

and

$$\gamma_i = -\frac{\mathbf{g_{i+1}} \cdot \mathbf{A} \cdot \mathbf{h_i}}{\mathbf{h_i} \cdot \mathbf{A} \cdot \mathbf{h_i}} = -\frac{\mathbf{g_{i+1}} \cdot (\mathbf{g_i} - \mathbf{g_{i+1}})/\lambda_i}{\mathbf{h_i} \cdot (\mathbf{g_i} - \mathbf{g_{i+1}})/\lambda_i} = \frac{\mathbf{g_{i+1}} \cdot \mathbf{g_{i+1}}}{\mathbf{h_i} \cdot \mathbf{g_i}} = \frac{\mathbf{g_{i+1}} \cdot \mathbf{g_{i+1}}}{\mathbf{g_i} \cdot \mathbf{g_i}} \tag{26}$$

which are exactly the same forms as in the CG algorithm in (14) and (12).

We can also prove that

$$\mathbf{h_i} \cdot \mathbf{g_k} = \mathbf{0} \qquad \forall\, 0 \leq i < k \leq m \tag{27}$$

as we expect, i.e. the gradient of the function at $\mathbf{x_k}$ is orthogonal to all the previous searching directions. This can be seen with the following

$$\mathbf{h_i} \cdot \mathbf{g_k} = \mathbf{h_i} \cdot (\mathbf{g_{k-1}} - \lambda_{k-1} \mathbf{A} \cdot \mathbf{h_{k-1}}) = \mathbf{h_i} \cdot \mathbf{g_{k-1}} = \ldots = \mathbf{h_i} \cdot \mathbf{g_{i+1}} = 0 \tag{28}$$

To summarise we have constructed two sequences of vectors $\mathbf{h_i}, \mathbf{g_i}$ in such a way that the former are *conjugate directions* and the latter are the gradients of the quadratical form at the line minima along the directions $\mathbf{h_i}$. These gradients are orthogonal to all previous searching directions, ensuring that the previous minimisation steps are not spoiled. In this way the maximum number of minimisation steps is equal to the dimensionality of the space.

2 Eigensystems, iterative methods

Now we discuss the second topic of these notes, i.e the eigenvalue problem

$$\mathbf{A} \cdot \mathbf{x} = \lambda \mathbf{x}. \tag{29}$$

We want to see how we can find only a few of the eigenvectors without diagonalising the whole matrix.

This problem arises in the solution of the one particle Schroedinger equations

$$H\psi_i = \epsilon_i \psi_i \tag{30}$$

We have seen that the wave-functions ψ_i can be expressed as linear combinations of basis functions,

$$\psi_i = \sum_{j=1}^{N} c_{ij}\phi_j \tag{31}$$

so that the Schroedinger equation 30 can be rewritten in the form

$$\mathbf{H} \cdot \mathbf{c}_i = \epsilon_i \mathbf{c}_i \tag{32}$$

with $H_{kl} = \phi_k \cdot H\phi_l$ and $\mathbf{c}_i = (c_{i1}, c_{i2}, \dots, c_{iN})$. The eigenvectors and the eigenvalues of $\mathbf{H}$ provide the solution of (32). Of course these are only an approximation to the true solutions of (30), since the expansion (31) corresponds to a projection of the Hilbert space of the eigenfunction of H onto a 'working space' spanned by the basis wave-functions $\{\phi_i\}$. The accuracy of the solution of Eq. (32) depends on how close the 'working space' is to the Hilbert space (or rather to the portion of Hilbert space where the ψ live). If one uses a plane wave basis set this is completely determined by the number of plane waves included in the basis. In a typical solid state problem the number of plane waves needed to have results within sufficient accuracy is of the order of a few hundreds per atom, depending on the atom type. However, one only needs a few wave-functions per atom to solve the problem (30), roughly half the number of electrons. Take silicon for example, and suppose you want to do a calculation with a cell containing 100 atoms. Each silicon atom has 4 valence electrons (if one uses the pseudo-potential approximation, where the core electrons are frozen and are not included in th calculations), so we need two wave-functions per atom, i.e. a total of 200 wave-functions to accommodate all the electrons. We may want to include a few more to have informations on the excited states as well. In any case, the total number of wave-functions needed is 200 or a few more. The total number of plane waves, however, is about 10000, i.e. 50 times more. To solve the problem exactly we need to diagonalise the matrix $\mathbf{H}$. The diagonalisation of a matrix is a N^3 process, and since we only need a number of eigenvectors $n << N$, it is not wise to diagonalise the whole matrix and retain only the lowest n eigenvectors and eigenvalues. It is advisable to look for an algorithm capable of finding only the lowest eigenvalues and eigenvectors *without* diagonalising the whole matrix. We discuss an algorithm to do so in what follows.

One important method for the approximate solution of the Schroedinger equation is the *variational method.* Define the functional

$$E[\psi] = \frac{\psi \cdot H\psi}{\psi \cdot \psi} \tag{33}$$

it is easy to show that $E[\psi]$ is at its *minimum* when and only when ψ is the ground state wave-function. Moreover, the functional $E[\psi]$ is stationary for each eigenstate of the Hamiltonian. This is a very powerful method, because it allows to put upper bounds to the ground state energy.

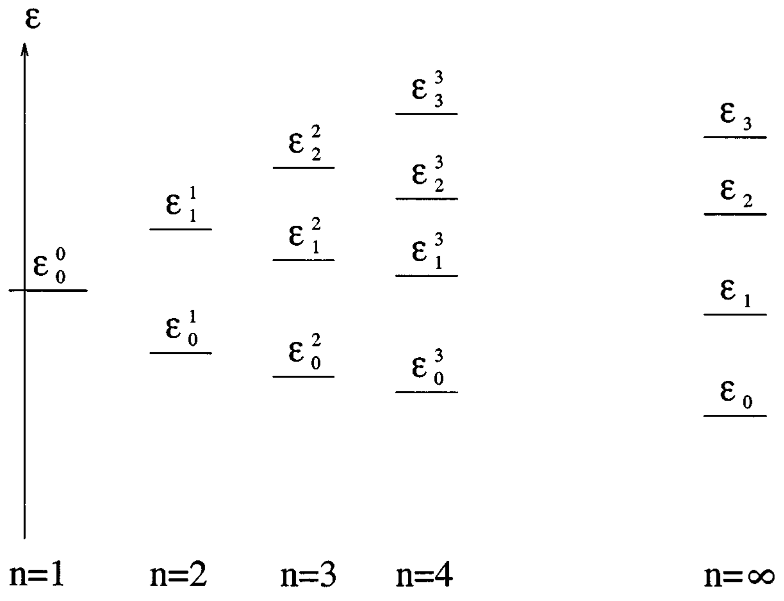

Figure 1: Approximate eigenvalues of the Hamiltonian H, each ϵ_i^n is an upper bound on the corresponding exact eigenvalue ϵ_i

To solve Equation (30) we need not just the ground state, i.e. the lowest eigenvalue and the lowest eigenstate, but a number of eigenstates, in the example of 100 silicon atoms the number of states needed were about 200. There is a remarkable result, known and the *Hylleraas-Undheim theorem*, which is a generalisation of the variational principle to excited states. Suppose that you construct a number of linearly independent trial wave-functions $\chi_1, \dots, \chi_n$. For simplicity, let us also assume that these are orthonormal (if they are not orthonormal they can be always orthonormalised). With these functions construct the $n \times n$ matrix $\mathbf{H}^n$ such that

$$H_{ij}^n = \chi_i \cdot H\chi_j \tag{34}$$

Now diagonalise the matrix and call $\mathbf{c}_0^n, \dots \mathbf{c}_{n-1}^n, \epsilon_0^n, \dots \epsilon_{n-1}^n$ the eigenvectors and the eigenvalues respectively, where $\mathbf{c}_i^n = (c_{1i}^n, c_{2i}^n, \dots, c_{ni}^n)$, and form the linear combinations

$$\psi_i^n = \sum_{j=1}^{n} c_{ij}^n \chi_j; \qquad i = 0, \dots, n-1 \tag{35}$$

The variational principle ensures that ϵ_0^n is an upper bound for the true ground state energy ϵ_0. It may be shown that each of the eigenvalues ϵ_i^n is an upper bound of the corresponding exact eigenvalue ϵ_i. The functions ψ_i^n are the best approximation for the exact eigenstates $\psi_0, \dots \psi_{n-1}$. Moreover, if we now add one trial wave-function to the original set, so that we have $\chi_1, \dots, \chi_n, \chi_{n+1}$ and they are all linearly independent, the *Hylleraas-Undheim theorem* states that the new eigenvalues $\epsilon_0^{n+1}, \dots, \epsilon_n^{n+1}$ are separated from the 'old' eigenvalues i.e. $\epsilon_0^{n+1} \le \epsilon_0^n, \dots, \epsilon_{n-1}^{n+1} \le \epsilon_{n-1}^n$. The new eigenstates

$$\psi_i^{n+1} = \sum_{j=1}^{n+1} c_{ij}^{n+1} \chi_j; \qquad i = 0, \dots, n \tag{36}$$

are now better approximations for the true eigenstates of the Hamiltonian H. A schematic picture of this scheme is given in Fig. (1) A proof of this theorem can be found in J.K. MacDonald Phys. Rev. **43**, 830 (1933).

This is a robust algorithm, since by increasing the size of the basis set $\{\chi\}$ one can find the lowest n eigenvalues of (32) with arbitrary precision. The following is a scheme of the algorithm.

Davidson Algorithm

1 Set a convergence threshold δ for the eigenvalues. Choose n trial orthonormal wave-functions $\chi_1, \dots \chi_n$, set $n_b = n, n_{notconv} = n$, construct the matrix $H_{ij}^{n_b} = \chi_i \cdot H\chi_j$ and set $k = 0$.

2 Set $\epsilon_i^k = H_{ii}^{n_b}$, and set $c_{i,j}^k = \delta_{ij}$.

3 Set $k = k+1$. Construct $\chi_{n_b+1} = G(\epsilon_0^k)(H - \epsilon_0^k)\psi_1^k, \dots, \chi_{n_b+n_{notconv}} = G(\epsilon_{n_{notconv}}^k)(H - \epsilon_{n_{notconv}}^k)\psi_{n_{notconv}}^k$, where $G(\epsilon) = \frac{1}{H_{diag}-\epsilon}$ is a *preconditioning* (see below) with H_{diag} the diagonal part of the Hamiltonian, and $\psi_i^k = \sum_{j=1}^{n_b} c_{ij}^k \chi_j$. Orthonormalise the new basis set $\{\chi_1, \dots, \chi_{n_b}, \chi_{n_{n_b}+1}, \dots, \chi_{n_{n_b+n_{notconv}}}\}$. Set $n_b = n_b + n_{notconv}$ and construct $H_{ij}^{n_b} = \chi_i \cdot H\chi_j$.

4 Diagonalise $\mathbf{H}^{n_b}$, let $\mathbf{c}_0^k, \dots \mathbf{c}_{n_b-1}^k$ and $\epsilon_0^k, \dots \epsilon_{n_b-1}^k$ the eigenvectors and eigenvalues respectively.

5 Check the convergence of the eigenvalues:

- set $n_{notconv} = 0$
- for $l = 0, \dots, n-1$ do
- if $|\epsilon_l^k - \epsilon_l^{k-1}| > \delta$ then $n_{notconv} = n_{notconv} + 1$

6 If $n_{notconv} = 0$ construct the linear combinations

$$\psi_i^k = \sum_{j=1}^{n_b} c_{ij}^k \chi_j; \qquad i = 0, \dots, n-1 \tag{37}$$

end exit, else goto step 3.

The extra trial wave-functions in step 3 could in principle be chosen at random, but it is better to construct them so that they are as close as possible to 'what is missing' to the exact wave-functions. The wave-function ψ_i^k can be written as a linear combination of the exact wave-function ψ_i and a part orthogonal to it $\psi_i^\perp$; $\chi_i = \alpha\psi_i + \beta\psi_i^\perp$, with α and β two appropriate complex numbers, and $\psi_i \cdot \psi_i^\perp = 0$. Since this wave-function is hopefully close to the exact wave-function the coefficient β will be a small number. The difference between the approximate eigenvalue ϵ_i^k and the exact eigenvalue ϵ_i is quadratic in β:

$$\begin{aligned}\epsilon_i^k - \epsilon_i = \delta\epsilon_i^k = \psi_i^k \cdot H\psi_i^k - \psi_i \cdot H\psi_i &= [(\alpha\psi_i + \beta\psi_i^\perp) \cdot H(\alpha\psi_i + \beta\psi_i^\perp)] - \psi_i \cdot H\psi_i = \\ &= (\alpha^2 - 1)\psi_i \cdot H\psi_i + \alpha\beta(\psi_i^\perp \cdot H\psi_i + c.c.) + \beta^2\psi_i^\perp \cdot H\psi_i^\perp = \\ &= \beta^2\epsilon_i + \alpha\beta(\epsilon_i\psi_i^\perp \cdot \psi_i + c.c) + \beta^2\psi_i^\perp \cdot H\psi_i^\perp = \mathrm{o}(\beta^2),\end{aligned} \tag{38}$$

and we have:

$$(H - \epsilon_i^k)\psi_i^k = (H - \epsilon_i - \delta\epsilon_i^k)(\alpha\psi_i + \beta\psi_i^\perp) = (H - \epsilon_i)\beta\psi_i^\perp - \mathrm{o}(\delta\epsilon_i^k). \tag{39}$$

If we knew the exact inverse Hamiltonian we could find the correcting vector $\beta\psi_i^\perp$ at once, neglecting term or order $\delta\epsilon_i^k$ which are quadratic in β. We don't know the inverse Hamiltonian so the best we can do is to use an approximate one which in our case is chosen to be the inverse of its diagonal part: $\frac{1}{H-\epsilon_i} \approx \frac{1}{H_{diag}-\epsilon_i^k}$.

This algorithm is not used in practice as it stands, since in principle there is no limit to the growing of the basis set of trial wave-functions, and one would like to avoid that. The algorithm is modified slightly as follows. One decides a maximum size of the matrix, usually this is taken to be 4 times the number of eigenstates needed, $n_{bmax} = d \times n, d = 4$. Once the number of trial wave-functions becomes bigger than that the first n trial wave-functions are substituted with the actual lower n eigenstates, $\chi_1 = \psi_0^k, \ldots, \chi_n = \psi_{n-1}^k$, n_b is set equal to n and the iterative process is restarted. In practice the 6th step of the algorithm is substituted with the following:

6 If $n_{notconv} = 0$ or $n_b > n_{bmax}$ construct the linear combinations

$$\psi_i^k = \sum_{j=1}^{n_b} c_{ij}^k \chi_j; \qquad i = 0, \ldots, n-1. \tag{40}$$

If $n_{notconv} = 0$ exit, else set $n_b = n$, $\chi_i = \psi_i^k$, $i = 0, \ldots, n-1$, and goto step 3.

Typically one only needs a few iterations to get the eigenvalues converged within the required accuracy.