Domain Decomposition and Electronic Structure Computations: A Promising Approach

G. Bencteux[1,4], M. Barrault[1], E. Cancès[2,4], W. W. Hager[3], and C. Le Bris[2,4]

[1] EDF R&D, 1 avenue du Général de Gaulle, 92141 Clamart Cedex, France
{guy.bencteux,maxime.barrault}@edf.fr

[2] CERMICS, École Nationale des Ponts et Chaussées, 6 & 8, avenue Blaise Pascal, Cité Descartes, 77455 Marne-La-Vallée Cedex 2, France,
{cances,lebris}@cermics.enpc.fr

[3] Department of Mathematics, University of Florida, Gainesville, FL 32611-8105, USA, hager@math.ufl.edu

[4] INRIA Rocquencourt, MICMAC project, Domaine de Voluceau, B.P. 105, 78153 Le Chesnay Cedex, FRANCE

Summary. We describe a domain decomposition approach applied to the specific context of electronic structure calculations. The approach has been introduced in [BCH06]. We survey here the computational context, and explain the peculiarities of the approach as compared to problems of seemingly the same type in other engineering sciences. Improvements of the original approach presented in [BCH06], including algorithmic refinements and effective parallel implementation, are included here. Test cases supporting the interest of the method are also reported.

It is our pleasure and an honor to dedicate this contribution to Olivier Pironneau, on the occasion of his sixtieth birthday. With admiration, respect and friendship.

1 Introduction and Motivation

1.1 General Context

Numerical simulation is nowadays an ubiquituous tool in materials science, chemistry and biology. Design of new materials, irradiation induced damage, drug design, protein folding are instances of applications of numerical simulation. For convenience we now briefly present the context of the specific computational problem under consideration in the present article. A more detailed, mathematically-oriented, presentation is the purpose of the monograph [CDK03] or of the review article [LeB05].

For many problems of major interest, empirical models where atoms are represented as point particles interacting with a parameterized force-field are adequate models. On the other hand, when electronic structure plays a role in the phenomenon under consideration, an explicit quantum modelling of the electronic wavefunctions is required. For this purpose, two level of approximation are possible.

The first category is the category of *ab initio* models, which are general purpose models that aim at solving sophisticated approximations of the Schrödinger equation. Such models only require the knowledge of universal constants and require a, ideally null but practically limited, number of adjustable parameters. The most commonly used models in this category are Density Functional Theory (DFT) based models and Hartree–Fock type models, respectively. Although these two families of models have different theoretical grounding, they share the same mathematical nature. They are *constrained minimization problems*, of the form

$$\inf \left\{ E(\psi_1, \ldots, \psi_N),\ \psi_i \in H^1(\mathbb{R}^3),\ \int_{\mathbb{R}^3} \psi_i \psi_j = \delta_{ij},\ \forall 1 \le i, j \le N \right\} \tag{1}$$

The functions ψ_i are called the *molecular orbitals* of the system. The energy functional E, which of course depends on the model employed, is parametrized by the charge and position of the nuclei of the system under consideration. With such models, systems with up to 10^4 electrons can be simulated.

Minimization problems of the type (1) are not approached by minimization algorithms, mainly because they are high-dimensional in nature. In contrast, the numerical practice consists in solving their Euler–Lagrange equations, which are nonlinear eigenvalue problems. The current practice is to iterate on the nonlinearity using fixed-point type algorithms, called in this framework *Self Consistent Field* iterations, with reference to the mean-field nature of DFT and HF type models.

The second category of models is that of semi-empirical models, such as Extended Hückel Theory based and tight-binding models, which contain additional approximations of the above DFT or HF type models. They consist in solving linear eigenvalue problems. State-of-the-art simulations using such models address systems with up to 10^5–10^6 electrons.

Finite-difference schemes may be used to discretize the above problems. They have proved successful in some very specific niches, most of them related to solid-state science. However, in an overwhelming number of contexts, the discretization of the nonlinear or linear eigenvalue problems introduced above is performed using a Galerkin formulation. The molecular orbitals ψ_i are developed on a Galerkin basis $\{\chi_i\}_{1 \le i \le N_b}$, with size $N_b > N$, the number of electrons in the system. Basis functions may be plane waves. This is often the case for solid state science applications and then N_b is very large as compared to N, typically one hundred times as large or more. They may also be *localized* functions, that is compactly supported functions or exponentially decreasing functions. Such basis sets correspond to the so-called *Linear Combination of Atomic Orbitals* (LCAO) approach. Then the dimension of the basis set needed to reach the extremely demanding accuracy required for electronic calculation problems is surprisingly small. Such basis sets, typically in the spirit of spectral methods, or modal synthesis, are indeed remarkably efficient. The domain decomposition method described in the present article is restricted to the LCAO approach. It indeed strongly exploits the locality of the basis functions.

In both categories of models, linear or nonlinear, the elementary brick of the solution procedure is the solution to a (generalized) linear eigenvalue problem of the following form:

$$\begin{cases} Hc_i = \varepsilon_i S c_i, \qquad \varepsilon_1 \leq \ldots \leq \varepsilon_N \leq \varepsilon_{N+1} \leq \ldots \leq \varepsilon_{N_b}, \\ c_i^t S c_j = \delta_{ij}, \\ D_\star = \displaystyle\sum_{i=1}^{N} c_i c_i^t. \end{cases} \tag{2}$$

The matrix H is a $N_b \times N_b$ symmetric matrix, called the *Fock matrix.* When the linear system above is one iteration of a nonlinear cycle, this matrix is computed from the result of the previous iteration. The matrix S is a $N_b \times N_b$ symmetric positive definite matrix, called the *overlap* matrix, which depends only on the basis set used (it corresponds to the mass matrix in the language of finite element methods).

One searches for the solution of (2), that is the matrix $D_\star$ called the *density matrix.* This formally requires the knowledge of the first N (generalized) eigenelements of the matrix H (in fact, we shall see below this statement is not exactly true).

The system of equations (2) is generally viewed as a generalized eigenvalue problem, and most of the computational approaches consist in solving the system via the computation of each individual vector c_i (discretizing the wavefunction ψ_i of (1)), using a direct diagonalization procedure.

1.2 Specificities of the Approaches for Large Systems

The procedure mentioned above may be conveniently implemented for systems of limited size. For large systems however, the solution procedure for the linear problem suffers from two computational bottlenecks. The first one is the need for assembling the Fock matrix. It *a priori* involves $O(N_b^3)$ operations in DFT models and $O(N_b^4)$ in HF models. Adequate approaches, which lower the complexity of this step, have been proposed. Fast multipole methods (see [SCh00]) are one instance of such approaches. The second practical bottleneck is the diagonalization step itself. This is the focus of the present contribution. Because of the possibly prohibitive $O(N_b^3)$ cost of direct diagonalization procedures, the so-called *alternatives to diagonalization* have been introduced. The method introduced in the present contribution aims at competing with such methods, and eventually outperforming them. With a view to understanding the problem under consideration, let us briefly review some peculiarities of electronic structure calculation problems.

The situation critically depends on the type of basis set employed. With plane wave basis sets, the number N of eigenelements to determine can be considered as small, compared to the size N_b of the matrix F ($N_b \sim 100N$). Then, iterative diagonalization methods, based on the inverse power paradigm, are a natural choice. In contrast, in the case of localized basis sets we deal with in this article, N_b varies from 2 to 10 times N. In any case it remains strictly proportional to N. Hence, problem (2) can be rephrased as follows: identify say one half of the eigenelements of a given matrix. This makes the problem very specific as compared to other linear eigenvalue problems encountered in other fields of the engineering sciences (see [AHL05, HLe05], for instance). The sparsity of the matrices in the present context is another peculiarity of the problem. Although the matrices H and S are sparse for large molecular systems, they are not as sparse as the stiffness and mass matrices usually encountered when using finite difference or finite element methods. For example, the bandwidth of H and S is of the order of 10^2 in the numerical examples reported in Section 5.

1.3 Alternative Methods Towards Linear Scaling

In addition to the above mentioned peculiarities, a crucial specificity of problem (2) is that the eigenelements do not need to be explicitly identified. As expressed by the last line of (2), only the knowledge of the density matrix $D_\star$ is required, both for the evaluation of the Fock operator associated to the next iteration, in a nonlinear context, and for the evaluation of relevant output quantities, in the linear context or at the last step of the iteration loop.

From a geometrical viewpoint, $D_\star$ is the S-orthogonal projector (in the sense that $D_\star S D_\star = D_\star$ and $D_\star^t = D_\star$) on the vector subspace generated by the eigenvectors associated with the lowest N eigenvalues of the generalized eigenvalue problem $Hc = \varepsilon Sc$.

Equations (2) define the orthogonal projector (in S dot product) onto the subspace spanned by the eigenvectors associated to the N lowest eigenvalues.

The above elementary remark is the bottom line for the development of the alternative to diagonalization methods, also often called *linear scaling* methods because their claimed purpose is to reach a linear complexity of the solution procedure (either in terms of N the number of electrons, or N_b the dimension of the basis set). For practical reasons, which will not be further developed here, such methods assume that:

(H1) The matrices H and S are sparse, in the sense that, for large systems, the number of non-zero coefficients scales as N. This assumption is not restrictive. In particular, it is automatically satisfied for DFT and HF models as soon as the basis functions are localized;

(H2) The matrix $D_\star$ built from the solution to (2) is also sparse. This condition seems to be fulfilled as soon as the relative gap

$$\gamma = \frac{\varepsilon_{N+1} - \varepsilon_N}{\varepsilon_{N_b} - \varepsilon_1}. \tag{3}$$

deduced from the solution of (2) is large enough. This observation can be supported by qualitative physical arguments [Koh96], but has seemingly no mathematical grounding to date (see, however, [Koh59]).

State-of-the-art surveys on such methods are [BMG02, Goe99]. One of the most commonly used linear scaling method is the Density Matrix Minimisation (DMM) method [LNV93].

2 A New Domain Decomposition Approach

Our purpose is now to expose a method, based on the *domain decomposition* paradigm, which we have recently introduced in [BCH06], and for which we also consider a setting where the above two assumptions are valid. Although still in its development, we have good hope that this approach will outperform existing ones in a near future. Preliminary test cases support this hope.

The approach described below is not the first occurrence of a method based on a "geographical" decomposition of the matrix H in the context of quantum chemistry (see, e.g., [YLe95]). A significant methodological improvement is however fulfilled with the present method. To the best of our knowledge, existing methods in the context of electronic calculations that may be recast as domain decomposition methods only consist of local solvers complemented by a crude global step. Our method seems to be the first one really exhibiting the local/global paradigm in the spirit of methods used in other fields of the engineering sciences.

In the following, we expose and make use of the method on one-dimensional systems, typically nanotubes or linear hydrocarbons. Generalizations to three-dimensional systems do not really bring up new methodological issues. They are however much more difficult in terms of implementation.

For simplicity, we now present our method assuming that $S = I_{N_b}$, i.e. that the Galerkin basis $\{\chi_i\}_{1 \le i \le N_b}$ is orthonormal. The extension of the method to the case when $S \ne I_{N_b}$ is straightforward. The space $\mathcal{M}^{k,l}$ denotes the vector space of the $k \times l$ real matrices.

Let us first notice that a solution $D_\star$ of (2) reads

$$D_\star = C_\star C_\star^t \tag{4}$$

where $C_\star$ is a solution to the minimization problem

$$\inf \left\{ \mathrm{Tr}(HCC^t), \quad C \in \mathcal{M}^{N_b,N}(\mathbb{R}),\ C^t S C = I_N \right\}. \tag{5}$$

Our approach consists in solving an *approximation* of problem (5). The latter is obtained by minimizing the exact energy $\mathrm{Tr}(HCC^t)$ on the set of the matrices C that have the block structure displayed on Figure 1 and satisfy the constraint $C^t C = I_N$.

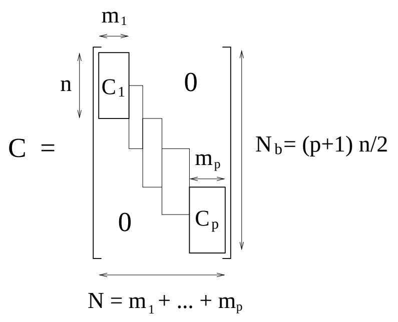


Fig. 1. Block structure of the matrices C.

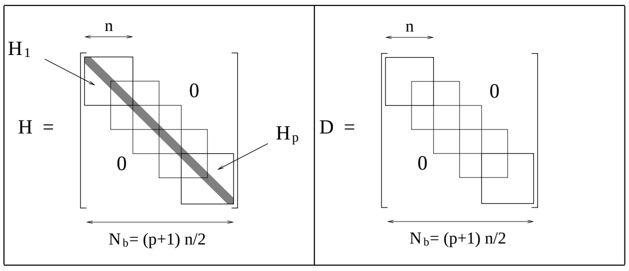


Fig. 2. Block structure of the matrices H and D.

A detailed justification of the choice of this structure is given in [BCH06]. Let us only mention here that the decomposition is suggested from the localization of electrons and the use of a localized basis set. Note that each block overlaps only with its first neighbors. Again for simplicity, we expose the method in the case where overlapping is exactly $n/2$, but it could be any integer smaller than $n/2$.

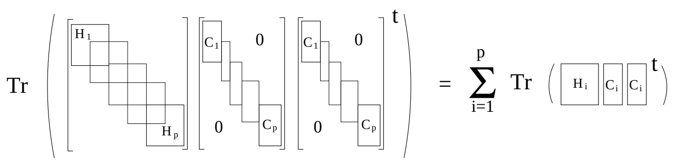
The resulting minimization problem can be recast as

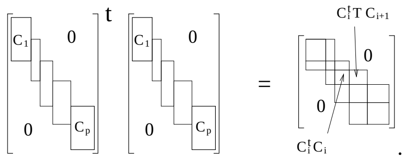
$$\inf \left\{ \sum_{i=1}^{p} \mathrm{Tr}\left(H_i C_i C_i^t\right),\ C_i \in \mathcal{M}^{n,m_i}(\mathbb{R}),\ m_i \in \mathbb{N},\ C_i^t C_i = I_{m_i}\ \forall 1 \le i \le p, \right.$$
$$\left. C_i^t T C_{i+1} = 0 \quad \forall\, 1 \le i \le p-1, \quad \sum_{i=1}^{p} m_i = N \right\}. \quad (6)$$

In the above formula, $T \in \mathcal{M}^{n,n}(\mathbb{R})$ is the matrix defined by

$$T_{kl} = \begin{cases} 1 & \text{if } k - l = \frac{n}{2}, \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

and $H_i \in \mathcal{M}^{n,n}(\mathbb{R})$ is a symmetric submatrix of H (see Figure 2). Indeed, and





In this way, we replace the $\frac{N(N+1)}{2}$ *global* scalar constraints $C^tC = I_N$ involving vectors of size N_b, by the $\sum_{i=1}^{p} \frac{m_i(m_i+1)}{2}$ *local* scalar constraints $C_i^tC_i = I_{m_i}$ and the $\sum_{i=1}^{p-1} m_i m_{i+1}$ *local* scalar constraints $C_i^tTC_{i+1} = 0$, involving vectors of size n. We would like to emphasize that we can only obtain in this way a basis of the vector space generated by the lowest N eigenvectors of H. This is the very nature of the method, which consequently cannot be applied for the search for the eigenvectors themselves.

Before we describe in details the procedure employed to solve the Euler–Lagrange equations of (6) in a greater generality, let us consider, for pedagogic purpose, the following oversimplified problem:

$$\inf \left\{ \langle H_1Z_1, Z_1\rangle + \langle H_2Z_2, Z_2\rangle,\ Z_i \in \mathbb{R}^{N_b},\ \langle Z_i, Z_i\rangle = 1,\ \langle Z_1, Z_2\rangle = 0 \right\}. \quad (8)$$

We have denoted by $\langle \cdot, \cdot \rangle$ the standard Euclidean scalar product on $\mathbb{R}^{N_b}$.

Problem (8) is not strictly speaking a particular occurence of (6), but it shows the same characteristics and technical difficulties: a separable functional is minimized, there are constraints on variables of each term and there is a cross constraint between the two terms.

The bottom line for our decomposition algorithm is to attack (8) as follows. Choose (Z_1^0, Z_2^0) satisfying the constraints and construct the sequence $(Z_1^k, Z_2^k)_{k\in\mathbb{N}}$ by the following iteration procedure. Assume (Z_1^k, Z_2^k) is known, then

Local step Solve

$$\begin{cases} \widetilde{Z}_1^k = \operatorname{arginf} \left\{ \langle H_1Z_1, Z_1\rangle,\ Z_1 \in \mathbb{R}^{N_b},\ \langle Z_1, Z_1\rangle = 1,\ \langle Z_1, Z_2^k\rangle = 0 \right\}, \\ \widetilde{Z}_2^k = \operatorname{arginf} \left\{ \langle H_2Z_2, Z_2\rangle,\ Z_2 \in \mathbb{R}^{N_b},\ \langle Z_2, Z_2\rangle = 1,\ \langle \widetilde{Z}_1^k, Z_2\rangle = 0 \right\}; \end{cases} \quad (9)$$

Global step Solve

$$\alpha^* = \operatorname{arginf} \left\{ \langle H_1Z_1, Z_1\rangle + \langle H_2Z_2, Z_2\rangle,\ \alpha \in \mathbb{R} \right\} \quad (10)$$

where

$$Z_1 = \frac{\widetilde{Z}_1^k + \alpha \widetilde{Z}_2^k}{\sqrt{1+\alpha^2}}, \quad Z_2 = \frac{-\alpha \widetilde{Z}_1^k + \widetilde{Z}_2^k}{\sqrt{1+\alpha^2}}, \tag{11}$$

and set

$$Z_1^{k+1} = \frac{\widetilde{Z}_1^k + \alpha^* \widetilde{Z}_2^k}{\sqrt{1+(\alpha^*)^2}}, \quad Z_2^{k+1} = \frac{-\alpha^* \widetilde{Z}_1^k + \widetilde{Z}_2^k}{\sqrt{1+(\alpha^*)^2}}. \tag{12}$$

This algorithm operates at two levels: a fine level where two problems of dimension N_b are solved (rather than one problem of dimension $2N_b$); a coarse level where a problem of dimension 2 is solved.

The local step monotonically reduces the objective function, however, it may not converge to the global optimum. The technical problem is that the Lagrange multipliers associated with the constraint $\langle Z_1, Z_2 \rangle = 0$ may converge to different values in the two subproblems associated with the local step. The global step again reduces the value of the objective function since $\widetilde{Z}_1^k$ and $\widetilde{Z}_2^k$ are feasible in the global step. The combined algorithm (local step + global step) therefore makes the objective function monotonically decrease. The simple case $H_1 = H_2$ is interesting to consider. First, if the algorithm is initialized with $Z_2^0 = 0$ in the first line of (9), it is easily seen that the local step is sufficient to converge to the global minimizer, in one single step. Second, it has been proved in [Bar05] that for a more general initial guess and under some assumption on the eigenvalues of the matrix H_1, this algorithm globally converges to an optimal solution of (8). Ongoing work [BCHL07] aims at generalizing the above proof when the additional assumption on eigenvalues is omitted. The analysis of the convergence in the case $H_1 \neq H_2$ is a longer term goal.

3 The Multilevel Domain Decomposition (MDD) Algorithm

We define, for all p-tuple $(C_i)_{1\le i\le p}$,

$$\mathcal{E}\Big((C_i)_{1\le i\le p}\Big) = \sum_{i=1}^{p} \mathrm{Tr}\left(H_i C_i C_i^t\right), \tag{13}$$

and set by convention

$$U_0 = U_p = 0. \tag{14}$$

It has been shown in [BCH06] that updating the block sizes m_i along the iterations is crucial to make the domain decomposition algorithm converge toward a good approximation of the solution to (5). It is however observed in practice that after a few iterations, the block sizes have converged (they do not vary in the course of the following iterations). This is why, for the sake of clarity, we have chosen to present here a simplified version of the algorithm where block sizes are held constant along the iterations. For a description of the complete algorithm with variable block sizes, we refer to [BCH06].

At iteration k, we have at hand a set of matrices $(C_i^k)_{1\leq i\leq p}$ such that $C_i^k \in \mathcal{M}^{n,m_i}(\mathrm{I\!R})$, $[C_i^k]^t C_i^k = I_{m_i}$, $[C_i^k]^t T C_{i+1}^k = 0$. We now explain how to compute the new iterate $(C_i^{k+1})_{1\leq i\leq p}$.

Step 1: **Local fine solver.**

a) For each i, find

$$\inf\left\{\mathrm{Tr}\left(H_i C_i C_i^t\right),\ C_i \in \mathcal{M}^{n,m_i}(\mathrm{I\!R}),\ C_i^t C_i = I_{m_i},\right.\\ \left.[C_{i-1}^k]^t T C_i = 0, \quad C_i^t T C_{i+1}^k = 0\right\}. \quad (15)$$

This is done via diagonalization of the matrix H_i in the subspace

$$V_i^k = \left\{x \in \mathrm{I\!R}^n,\ \left[C_{i-1}^k\right]^t Tx = 0,\ x^t T C_{i+1}^k = 0\right\},$$

i.e. diagonalize $P_i^k H_i P_i^k$ where P_i^k is the orthogonal projector on V_i^k.

This provides (at least) $n - m_{i-1} - m_{i-1}$ real eigenvalues and associated orthonormal vectors $x_{i,j}^k$. The latter are T-orthogonal to the column vectors of C_{i-1}^k and C_{i+1}^k.

b) Collect the lowest m_i vectors $x_{i,j}^k$ in the $n \times m_i$ matrix $\widetilde{C}_i^k$.

Step 2: **Global coarse solver.** Solve

$$\mathcal{U}^* = \mathrm{arginf}\left\{f(\mathcal{U}),\ \mathcal{U} = (U_i)_i,\ \forall 1 \leq i \leq p-1,\ U_i \in \mathcal{M}^{m_{i+1},m_i}(\mathrm{I\!R})\right\}, \quad (16)$$

where

$$f(\mathcal{U}) = \mathcal{E}\left(\left(C_i(\mathcal{U})\left(C_i(\mathcal{U})^t C_i(\mathcal{U})\right)^{-\frac{1}{2}}\right)_i\right), \quad (17)$$

and

$$C_i(\mathcal{U}) = \widetilde{C}_i^k + T\widetilde{C}_{i+1}^k U_i\left([\widetilde{C}_i^k]^t T T^t \widetilde{C}_i^k\right) - T^t \widetilde{C}_{i-1}^k U_{i-1}^t\left([\widetilde{C}_i^k]^t T^t T \widetilde{C}_i^k\right). \quad (18)$$

Next set, for all $1 \leq i \leq p$,

$$C_i^{k+1} = C_i\left(\mathcal{U}^*\right)\left(C_i\left(\mathcal{U}^*\right)^t\ C_i\left(\mathcal{U}^*\right)\right)^{-1/2}. \quad (19)$$

Notice that in Step 1, the computations of each odd block is independant from the other odd blocks, and obviously the same for even blocks. Thus, we use here a red/black strategy.

In the global step, we perturb each variable by a linear combination of the adjacent variables. The matrices $\mathcal{U} = (U_i)_i$ in (16) play the same role as the real parameter α in the toy example, equation (10). The perturbation is designed so that the constraints are satisfied. However, our numerical experiments show that this is not exactly the case, in the sense that, for some i, $[C_i^{k+1}]^t[C_i^{k+1}]$ may presents coefficients as large as about 10^{-3}. All linear scaling algorithms have difficulties in ensuring this constraint. We should mention

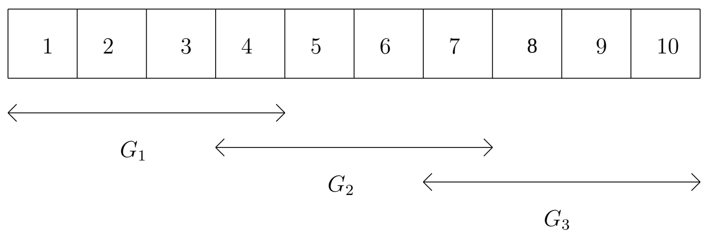


Fig. 3. Collection of $p = 10$ blocks into $r = 3$ groups.

```
Repeat until convergence:

1a. Local step on blocks: 1, 3, ..., $(2i+1)$, ...
1b. Local step on blocks: 2, 4, ..., $(2i)$, ...
2a. Global step on groups: $\{1,2\}$, $\{3,4\}$, ..., $\{2i-1,2i\}$, ...
2b. Global step on groups: $\{2,3\}$, $\{4,5\}$, ..., $\{2i,2i+1\}$, ...
```

Fig. 4. Schematic view of the algorithm in the case of 2-block groups ($r = 2$): tasks appearing on the same line are independent from one another. Order between steps 1a and 1b is reversed from on iteration to the other. The same holds for steps 2a and 2b.

here that in our case, the resulting deviation of C^tC from identity is small, C^tC being in any case block tridiagonal.

In practice, we reduce the computational cost of the global step, by again using a domain decomposition method. The blocks $(C_i)_{1\le i\le p}$ are collected in r overlapping groups $(G_l)_{1\le l\le r}$ as shown in Figure 3. As each group only overlaps with its first neighbors, problem (16) can be solved first for the groups (G_{2l+1}), next for the groups (G_{2l}). We have observed that the number of iterations of the outer loop (local step + global step) does not significantly increases when the 'exact' global step (16) is replaced by the approximate global step consisting in optimizing first the odd groups, then the even groups. The numerical results performed so far (see Section 5) tend to show that the resulting algorithm scales linearly with the system size.

A schematic view of the algorithm is provided in Figure 4.

One important point (not taken into account in [BCH06]) is that the Hessian of f enjoys a very specific structure. It is a sum of tensor products of square matrices of size m_i. For example, with two-block groups ($r = 2$), we have:

$$HU = \sum_{i=1}^{4} A^{(i)} U B^{(i)} \tag{20}$$

with $A^{(i)} \in \mathcal{M}^{m_2,m_2}(\mathbb{R})$ and $B^{(i)} \in \mathcal{M}^{m_1,m_1}(\mathbb{R})$. Consequently, it is possible to compute hessian-vector products, without assembling the hessian, in $\mathcal{O}(m_1\ m_2\ \max(m_1,m_2))$ elementary operations, instead of $\mathcal{O}(m_1^2\ m_2^2)$ with

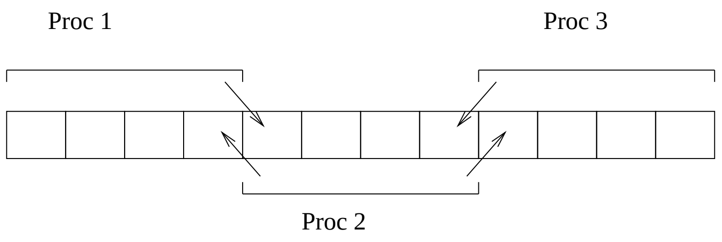


Fig. 5. Distribution of blocks over 3 processors. Arrows indicate the supplementary blocks a processor need to access.

a naive implementation. An additional source of acceleration is the fact that this formulation uses only matrix-matrix products. Efficient implementations of matrix-matrix products, taking advantage of higher numbers of floating point operations per memory access, are available in the BLAS 3 library (see, for instance, [PeA04]). This makes Newton-like methods affordable: a good estimation of the Newton direction can be easily computed using an iterative method.

In the current version of our domain decomposition algorithm, the global step is solved approximatively by a single iteration of the Newton algorithm with initial guess $U_i = 0$, the Newton iteration being computed iteratively by means of the SYMMLQ algorithm [PSa75]. In a next future, we plan to test the efficiency of advanced first order methods such as the one described in [HaZ05]. No definite conclusions about the comparative efficiencies of the various numerical methods for performing the global step can be drawn yet.

4 Parallel Implementation

For parallel implementation, the single-program, multi-data (SPMD) model is used, with message passing techniques using the MPI library, which allows to maintain only one version of the code.

Each processor executes a single instance of the algorithm presented in Section (3) applied to a contiguous subset of blocks. Compared to the sequential version, additional data structures are introduced: each processor needs to access the matrices C_i and H_i corresponding to the last block of the processor located on its left and to the first block of the processor located on its right, as shown in Figure 5. These frontier blocks play the role of ghost nodes in classical domain decomposition without overlapping. For this reason, we will sometimes call them the ghost blocks.

The major part of the communications is performed between neighboring processors at the end of each step of the algorithm (i.e. of each line in the scheme displayed in Figure (4)), in order to update the ghost blocks. This occurs only four times per iteration and, as we will see in the next section, the sizes of the exchanged messages are moderate.

Collective communications are needed to compute the current value of the function f and to check that the maximum deviation from orthogonality remains acceptable. They are also needed to sort the eigenvalues of the different blocks in the local step, in the complete version of the algorithm, allowing variable block sizes (see [BCH06]). The important point is that the amount of data involved in the collective communications is small as well.

With this implementation we can use up to $nbloc/2$ processors. In order to efficiently use a larger number of processors, sublevels of parallelism should be introduced. For instance, each subproblem (15) (for a given i) can itself be parallelized.

Apart from the very small part of collective communications, the communication volume associated with each single processor remains constant irrespective of the number of blocks per processor and of the number of processors. We can thus expect a very good scalability, except for the situations when load balancing is strongly heterogeneous.

The implementation of the MDD algorithm described above can be easily extended to cover the case of $2D$ and $3D$ molecular systems.

5 Numerical Tests

This section is devoted to the presentation of the performances of the MDD algorithm on matrices actually arising in real-world applications of electronic structure calculations. The benchmark matrices are of the same type of those used in the reference paper [BCH06].

In a first subsection, we briefly recall how these matrices are generated and we provide some practical details on our implementation of the MDD algorithm. The computational performances obtained on sequential and parallel architectures, including comparisons with other methods (diagonalization and DMM), are discussed in the second and third subsections, respectively.

5.1 General Presentation

Three families of matrices corresponding to the Hartree–Fock ground state of some polymeric molecules are considered:

- Matrices of type $\mathcal{P}_1$ and $\mathcal{P}_2$ are related to COH-$(CO)_{n_m}$-COH polymeric chains, with interatomic Carbon-Carbon distances equal to 5 and 4 respectively;
- Matrices of type $\mathcal{P}_1$ are obtained with polyethylen molecules (CH_3-$(CH_2)_{n_m}$-CH_3) with physically relevant Carbon-Carbon distances.

The geometry of the very long molecules is guessed from the optimal distances obtained by geometry optimization (with constraints for $\mathcal{P}_1$ and $\mathcal{P}_2$) on moderate size molecules (about 60 Carbon atoms) and minimal basis sets. All these off-line calculations are performed using the GAUSSIAN package ([FTS98]).

Table 1. Localization parameters, block sizes and asymptotic gaps for the test cases.

	$\mathcal{P}_1$	$\mathcal{P}_2$	$\mathcal{P}_3$
Bandwith of S	59	79	111
Bandwith of H	99	159	255
n	130	200	308
q	50	80	126
Asymptotic gap (u.a.)	1.04×10^{-3}	3.57×10^{-3}	2.81×10^{-2}

It is then observed that the overlap matrix and Fock matrix obtained exhibit a periodic structure in their bulk. Overlap and Fock matrices for large size molecules can then be constructed using this periodicity property. For n_m sufficiently large, bulk periodicity is also observed in the density matrix. This property is used to generate reference solutions for large molecules.

Table 1 gives a synthetic view of the different structure properties of the three families of matrices under examination. The integer q stands for the overlap between two adjacent blocks (note that one could have taken $n = 2q$ if the overlap matrix S was equal to identity, but that one has to take $n > 2q$ in our case since $S \neq I$).

Initial guess generation is of crucial importance for any linear scaling method. The procedure in use here is in the spirit of the domain decomposition method:

1. A first guess of the block sizes is obtained by locating Z electrons around each nucleus of charge Z;
2. A set of blocks C_i is built from the lowest m_i (generalized) eigenvectors associated with the block matrices H_i and S_i (the block matrices H_i are introduced in Section 2; the block matrices S_i are defined accordingly);
3. These blocks are eventually optimized with the local fine solver of the MDD algorithm, including block size update (electron transfer).

Criteria for comparing the results

The quality of the results produced by the MDD and DMM methods is evaluated by computing two criteria. The first criterion is the relative energy difference $e_E = \dfrac{|E - E_0|}{|E_0|}$ between the energy E of the current iterate D and the energy E_0 of the reference density matrix $D_\star$. The second criterion is the semi-norm

$$e_\infty = \sup_{(i,j)\,\text{s.t.}\,|H_{ij}|\geq\varepsilon} \left| D_{ij} - [D_\star]_{ij} \right|, \tag{21}$$

with $\varepsilon = 10^{-10}$. The introduction of the semi-norm (21) is consistent with the cut-off on the entries of H (thus the value chosen for ε). Indeed, in most cases, the matrix D is only used for the calculations of various observables (in particular the electronic energy and the Hellman–Feynman forces), all of them of the form $\mathrm{Tr}(AD)$, where the matrix A shares the same pattern as H (see [CDK03] for details). The final result of the calculation is therefore insensitive to entries D_{ij} with indices (i, j) such that $|H_{ij}|$ is below some cut-off value.

In all the calculations presented below, the global step is performed with groups consisting of two blocks ($r = 2$), and the algorithm is therefore exactly that displayed in Figure 4.

5.2 Sequential Computations

The numerical results presented in this section have been obtained with a single 2.8 GHz Xeon processor.

Density matrices have been computed for a series of matrices H and S of types $\mathcal{P}_1$, $\mathcal{P}_2$, and $\mathcal{P}_3$, using (i) the MDD algorithm, (ii) a diagonalization procedure (the *dsbgv.f* routine from the LAPACK library), and (iii) the DMM method [LNV93]. The latter method belongs to the class of linear scaling algorithms. An important feature of the DMM method is that linear scaling is achieved through cut-offs on the matrix entries. We have chosen here a cut-off strategy based on *a priori* defined patterns, that may be suboptimal. Our implementation of DMM converges to a fairly good approximation of the exact density matrix and scales linearly, but the prefactor might possibly be improved by more refined cut-off strategies.

A detailed presentation of the comparison between the three methods is provided in [BCH06]. Our new approach for computing the Newton direction in the global step (see Section 3) further improves the efficiency of MDD: with the new implementation of MDD, and with respect to the former implementation reported on in [BCH06], CPU is divided by 2 for $\mathcal{P}_1$ type molecules, by 5 for $\mathcal{P}_2$, and by 10 for $\mathcal{P}_3$, and the memory required is now lower for MDD than for DMM. These results are shown for $\mathcal{P}_2$ in Figures 6 and 7. They clearly demonstrate that the MDD algorithm scales linearly with respect to the parameter n_m (in both CPU time and memory occupancy).

Let us also notice that for $\mathcal{P}_2$, the crossover point between diagonalisation and MDD (as far as CPU time is concerned) is now shifted to less than 2,000 basis functions.

5.3 Parallel Computations

We conclude with some tests of our parallel implementation of the MDD algorithm described in Section 4. These tests have been performed on a 8 node Linux cluster in dedicated mode, consisting of 8 biprocessor DELL Precision 450 (Intel(R) Xeon(TM) CPU 2.40GHz), with Gigabit Ethernet connections. They concern the polyethylen family $\mathcal{P}_3$, for which the size of each ghost block is about 150 Ko.

We only test here the highest level of parallelism of the MDD algorithm, consistently with the relatively low number of processors that have been used in this first study. We plan to test multilevel parallelism in a near future.

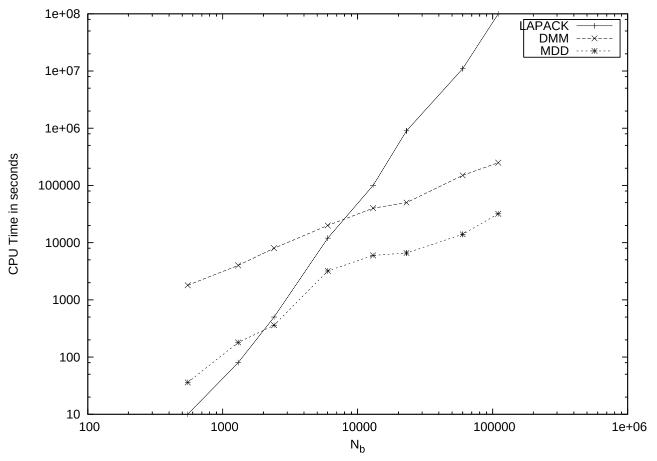


Fig. 6. Requested CPU time for computing the density matrix of a molecule of type $\mathcal{P}_2$ as a function of the number of basis functions.

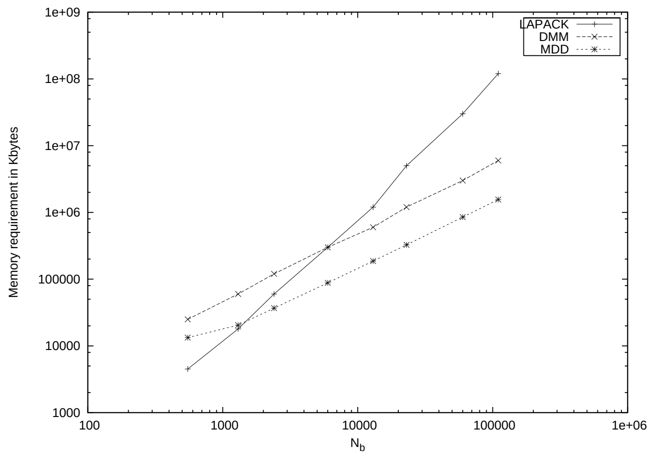


Fig. 7. Requested memory for computing the density matrix of a molecule of type $\mathcal{P}_2$ as a function of the number of basis functions.

In particular, the local step in each block, as well as the global step in each group, will be parallelized.

Tables 2 and 3 report on the speedup (ratio between the wall clock time with one processor and the wall clock time for several processors) and efficiency (ratio between the speedup and the number of processors) of our parallel MDD algorithm.

The scalability, namely the variation of the wall clock time when the number of processors and the size of the matrix proportionally grow, is reported in Table 4, for a molecule of type $\mathcal{P}_3$.

Table 2. Wall clock time as a function of the number of processors for a molecule of type $\mathcal{P}_3$, with $n_m = 3300$. 8 MDD iterations are necessary to achieve convergence up to 5×10^{-8} in energy and 3×10^{-3} in the density matrix (for the semi-norm (21)).

Number of processors	1	2	4	8	16
Wall clock time (s)	4300	2400	1200	580	360
Speedup		1.8	3.6	7.4	12
Efficiency		0.9	0.9	0.9	0.75

Table 3. Wall clock time as a function of the number of processors for a molecule of type $\mathcal{P}_3$, with $n_m = 13300$. 7 MDD iterations are necessary to achieve convergence up to 5×10^{-8} in energy and 3×10^{-3} in the density matrix (for the semi-norm (21)).

Number of processors	1	4	8	16
Wall clock time (s)	18460	4820	2520	1275
Speedup		3.8	7.3	14.5
Efficiency		0.96	0.92	0.91

Table 4. Scalability of the MDD algorithm for a molecule of type $\mathcal{P}_3$. The convergence thresholds are 2.5×10^{-7} in energy and 4×10^{-3} in density matrix (for the semi-norm (21)).

Number of processors	1	4	8	16
Wall clock time with 200 atoms per processor (s)	167	206	222	253
Wall clock time with 800 atoms per processor (s)	1249	1237	1257	1250

Note that the calculations reported in this article have been performed with minimal basis sets. It is the subject of ongoing works to test the efficiency of the MDD algorithm for larger basis sets.

Let us finally mention that our parallel implementation of the MDD algorithm allows to solve (2) for a polyethylen molecule with 106 530 atoms (372 862 basis functions) on 16 processors, in 90 minutes.

6 Conclusion and Perspectives

In its current implementation, the MDD algorithm allows to solve efficiently the linear subproblem for linear molecules (polymers or nanotubes). The following issues will be addressed in a near future:

- Still in the case of 1D systems, we will allow blocks to have more than two neighbors. This should increase the flexibility and efficiency of the MDD algorithm. For instance, this should render calculations with large basis sets including diffuse atomic orbitals affordable.

- We plan to implement the MDD algorithm in the framework of 2D and 3D molecular systems. Note that even with minimal overlap a given block has typically 8 neighbors in 2D and 26 neighbors in 3D.
- The MDD algorithm will be extended to the cases of the nonlinear Hartree–Fock and Kohn-Sham problems.
- The present version of the MDD algorithm is restricted to insulators (i.e. to matrices H with a sufficiently large gap). The possibility of extending the MDD methodology to cover the case of metallic systems is a challenging issue that will be studied.

Acknowledgments

EC and CLB acknowlegde financial support from the French Ministry for research under contract grant "Nouvelles Interfaces des Mathématiques" SIMUMOL, and from Electricité de France under contract EDF-ENPC. WH acknowledges support from US National Science Foundation under grants 0203370, 0620286, and 0619080.

References

[AHL05] P. Arbenz, U. L. Hetmaniuk, R. B. Lehoucq and R. S. Tuminaro. A comparison of eigensolvers for large-scale 3D modal analysis using AMG-preconditioned iterative methods. *Int. J. Numer. Meth. Engng*, 64:204–236, 2005.

[Bar05] M. Barrault. *Développement de méthodes rapides pour le calcul de structures électroniques*, thèse de l'Ecole Nationale des Ponts et Chaussées, 2005.

[BCH06] M. Barrault, E. Cancès, W. W. Hager, C. Le Bris. Multilevel domain decomposition for electronic structure calculations. *J. Comp. Phys.*, to appear.

[BCHL07] G. Bencteux, E. Cancès, W. W. Hager, C. Le Bris. Work in progress.

[BMG02] D. Bowler, T. Miyazaki and M. Gillan. Recent progress in linear scaling ab initio electronic structure theories. *J. Phys. Condens. Matter*, 14:2781–2798, 2002.

[CDK03] E. Cancès, M. Defranceschi, W. Kutzelnigg, C. Le Bris, and Y. Maday. *Computational Quantum Chemistry: a Primer*, In C. Le Bris, editor, *Handbook of Numerical Analysis, Special volume, Computational Chemistry, volume X*, North-Holland 2003.

[Goe99] S. Goedecker. Linear scaling electronic structure methods. *Rev. Mod. Phys.*, 71:1085–1123, 1999.

[HaZ05] W. Hager and H. Zhang. A new conjugate gradient method with guaranteed descent and an efficient line search. *SIAM Journal on Optimization*, 16:170–192, 2005.

[HLe05] U. L. Hetmaniuk and R. B. Lehoucq. Multilevel methods for eigenspace computations in structural dynamics. In *Proceedings of the 16th International Conference on Domain Decomposition Methods, Courant Institute, New York, January 12–15, 2005.*

[Koh59] W. Kohn. Analytic properties of Bloch waves and Wannier functions. *Phys. Rev.*, 115:809–821, 1959.

[Koh96] W. Kohn. Density functional and density matrix method scaling linearly with the number of atoms. *Phys. Rev. Lett.*, 76:3168–3171, 1996.

[LeB05] C. Le Bris. *Computational chemistry from the perspective of numerical analysis*, In Acta Numerica, Volume 14, 2005, pp. 363–444.

[LNV93] X.-P. Li, R. W. Nunes, and D. Vanderbilt. Density-matrix electronic structure method with linear system size scaling. *Phys. Rev. B*, 47:10891–10894, 1993.

[PeA04] W. P. Petersen and P. Arbenz. *Introduction to Parallel Computing.* Oxford University Press, 2004.

[PSa75] C. Paige and M. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM J. Numer. Anal.*, 12:617–629, 1975.

[SCh00] E. Schwegler and M. Challacombe. Linear scaling computation of the Fock matrix. *Theor. Chem. Acc.*, 104:344–349, 2000.

[YLe95] W. Yang and T. Lee. A density-matrix divide-and-conquer approach for electronic structure calculations of large molecules. *J. Chem. Phys.*, 163:5674, 1995.

[FTS98] M.J. Frisch, G.W. Trucks, H.B. Schlegel, G.E. Scuseria, M.A. Robb, J.R. Cheeseman, V.G. Zakrzewski, J.A. Montgomery, R.E. Stratmann, J.C. Burant, S. Dapprich, J.M. Millam, A.D. Daniels, K.N. Kudin, M.C. Strain, O. Farkas, J. Tomasi, V. Barone, M. Cossi, R. Cammi, B. Mennucci, C. Pomelli, C. Adamo, S. Clifford, J. Ochterski, G.A. Petersson, P.Y. Ayala, Q. Cui, K. Morokuma, D.K. Malick, A.D. Rabuck, K. Raghavachari, J.B. Foresman, J. Cioslowski, J.V. Ortiz, B.B. Stefanov, G. liu, A. Liashenko, P. Piskorz, I. Kpmaromi, G. Gomperts, R.L. Martin, D.J. Fox, T. Keith, M.A. Al-Laham, C.Y. Peng, A. Nanayakkara, C. Gonzalez, M. Challacombe, P.M.W. Gill, B.G. Johnson, W. Chen, M.W. Wong, J.L. Andres, M. Head-Gordon, E.S. Replogle and J.A. Pople. Gaussian 98 (Revision A.7). Gaussian Inc., Pittsburgh PA 1998.