Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational Physics 222 (2007) 86–109
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Multilevel domain decomposition for electronic structure calculations

M. Barrault [a,b,*], E. Cancès [b], W.W. Hager [c], C. Le Bris [b]

[a] EDF R&D, 1 avenue du Général de Gaulle, 92141 Clamart Cedex, France
[b] CERMICS, École Nationale des Ponts et Chaussées, 6 & 8, Avenue Blaise Pascal, Cité Descartes, 77455 Marne-La-Vallée Cedex 2, France
[c] Department of Mathematics, University of Florida, Gainesville, FL 32611-8105, USA

Received 25 October 2005; received in revised form 8 June 2006; accepted 28 June 2006
Available online 28 November 2006

Abstract

We introduce a new multilevel domain decomposition method (MDD) for electronic structure calculations within semi-empirical and density functional theory (DFT) frameworks. This method iterates between local fine solvers and global coarse solvers, in the spirit of domain decomposition methods. Using this approach, calculations have been successfully performed on several linear polymer chains containing up to 40,000 atoms and 200,000 atomic orbitals. Both the computational cost and the memory requirement scale linearly with the number of atoms. Additional speed-up can easily be obtained by parallelization. We show that this domain decomposition method outperforms the density matrix minimization (DMM) method for poor initial guesses. Our method provides an efficient preconditioner for DMM and other linear scaling methods, variational in nature, such as the orbital minimization (OM) procedure.

Keywords: Electronic structure calculations; Linear scaling method; Domain decomposition method; Computational chemistry; Order-N method

1. Introduction and motivation

A central issue in computational quantum chemistry is the determination of the electronic ground state of a molecular system. For completeness and self-consistency, we now briefly introduce the problem. In particular, we present it in a mathematical way.

* Corresponding author. Address: CERMICS, École Nationale des Ponts et Chaussées, 6 & 8, Avenue Blaise Pascal, Cité Descartes, 77455 Marne-La-Vallée Cedex 2, France.
E-mail addresses: maxime.barrault@edf.fr, barrault@cermics.enpc.fr (M. Barrault), cances@cermics.enpc.fr (E. Cancès), hager@math.ufl.edu (W.W. Hager), lebris@cermics.enpc.fr (C. Le Bris).

0021-9991/$ - see front matter © 2006 Published by Elsevier Inc.
doi:10.1016/j.jcp.2006.06.049

1.1. Standard electronic structure calculations

A molecular system is composed of N electrons, modelled quantum mechanically, and a given number of nuclei, the latter being considered as classical point-like particles clamped at known positions (Born–Oppenheimer approximation). We refer to [7] for a general mathematical exposition and to [18,25] for the chemical background. Determining the electronic ground state amounts to solving a time-independent Schrödinger equation in $\mathbb{R}^{3N}$. This goal is out of reach for large values of N. In fact it is already infeasible for values of N exceeding three or four, unless dedicated techniques are employed. Examples are stochastic-like techniques such as diffusion Monte-Carlo approaches, or emerging techniques, such as sparse tensor products techniques [23]. Approximations of the Schrödinger equation have been developed, such as the widely used *tight-binding*, *Hartree–Fock* and *Kohn–Sham* models. For these three models, the numerical resolution of a problem of the following type is required: given H and S, respectivement an $N_b \times N_b$ symmetric matrix and an $N_b \times N_b$ symmetric positive definite matrix (with $N_b > N$), compute a solution $D_\star$ of the problem

$$\begin{cases} Hc_i = \epsilon_i S c_i, \qquad \epsilon_1 \leqslant \cdots \leqslant \epsilon_N \leqslant \epsilon_{N+1} \leqslant \cdots \leqslant \epsilon_{N_b}, \\ c_i^t S c_j = \delta_{ij}, \\ D_\star = \sum_{i=1}^{N} c_i c_i^t. \end{cases} \tag{1.1}$$

Let us mention that most electronic structure calculations are performed with closed shell models [18], and that, consequently, the integer N in (1.1) then is the number of electron pairs. We remark that when S is the identity matrix, a solution $D_\star$ to (1.1) is a solution to the problem

$$\begin{cases} \text{Find the orthogonal projector on the space spanned by the } N \text{ eigenvectors} \\ \text{associated with the lowest } N \text{ eigenvalues of } H. \end{cases} \tag{1.2}$$

In (1.2), and throughout this article, the eigenvalues are counted with their multiplicities. The N eigenvectors c_i, called *generalized* eigenvectors in order to emphasize the presence of the matrix S, represent the expansion in a given Galerkin basis $\{\chi_i\}_{1 \leqslant i \leqslant N_b}$ of the N one-electron wavefunctions. The matrix H is a mean-field Hamiltonian matrix. For instance, for the Kohn–Sham model, we have

$$H_{ij} = \frac{1}{2} \int_{\mathbb{R}^3} \nabla \chi_i \cdot \nabla \chi_j + \int_{\mathbb{R}^3} V \chi_i \chi_j, \tag{1.3}$$

where V is a mean-field local potential. The matrix S is the overlap matrix associated with the basis $\{\chi_i\}_{1 \leqslant i \leqslant N_b}$:

$$S_{ij} = \int_{\mathbb{R}^3} \chi_i \chi_j. \tag{1.4}$$

In this article, we focus on the *linear combination of atomic orbitals* (LCAO) approach. This is a very efficient discretization technique, using localized basis functions $\{\chi_i\}$, compactly supported [29] or exhibiting a Gaussian fall-off [18].

It is important to emphasize what makes the electronic structure problem, discretized with the LCAO approach, specific as compared to other linear eigenvalue problems encountered in other fields of the engineering sciences (see [2,19] for instance). First, N_b is proportional to N, and not much larger than it (say $N_b \sim 2N$ to fix the ideas). Hence, the problem is not finding a few eigenvectors of the generalized eigenvalue problem (1.1). Second, although the matrices H and S are sparse for large molecular systems (see Section 1.2 for details), 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 4. Note that, in contrast, for plane wave basis set discretizations (which will not be discussed here), the parameter N_b is much larger than N (say $N_b \sim 100N$), the matrix S is the identity matrix and the matrix H est full. Third, and this is a crucial point, the output of the calculation is the matrix $D_\star$ and not the generalized eigenvectors c_i themselves. This is the fundamental remark allowing the construction of linear scaling methods (see Section 1.2).

A solution $D_\star$ of (1.1) is

$$D_\star = C_\star C_\star^t, \tag{1.5}$$

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

$$\inf\left\{\mathrm{Tr}(HCC^t), \quad C \in \mathscr{M}^{N_b,N}(\mathbb{R}),\ C^tSC = I_N\right\}. \tag{1.6}$$

Note that the energy functional $\mathrm{Tr}(HCC^t)$ can be given the more symmetric form $\mathrm{Tr}(C^tHC)$. Here and below, $\mathscr{M}^{k,l}$ denotes the vector space of the $k \times l$ real matrices. Notice that (1.6) has many minimizers: if $C_\star$ is a minimizer, so is $C_\star U$ for any orthogonal $N \times N$ matrix U. However, under the standard assumption that the Nth eigenvalue of H is strictly lower than the $(N+1)$th one, the matrix D^* defined by (1.5) does not in fact depend on the choice of the minimizer $C_\star$ of (1.6). Notice also that (1.1) are not the Euler–Lagrange equations of (1.6) but that any critical point of (1.6) is obtained from a solution of (1.1) by an orthogonal transformation of the columns of $C_\star = (c_1|\cdots|c_N)$.

The standard approach to compute $D_\star$ is to solve the generalized eigenvalue problem (1.1) and then construct $C_\star$ thus $D_\star$ by collecting the lowest N generalized eigenvectors of H. This approach is employed when the number N of electrons (or electron pairs) is not too large, say smaller than 10^3.

1.2. Linear scaling methods

One of the current challenges of Computational Chemistry is to lower the computational complexity N^3 of this solution procedure. A linear complexity N is the holy grail. There are various existing methods designed for this purpose. Surveys on such methods are [6,17]. Our purpose here is to introduce a new method, based on the *domain decomposition* paradigm. We remark that the method introduced here is not the first occurrence of a method based on a decomposition of the matrix H [30], but a significant methodological improvement is fulfilled with the present method. To the best of our knowledge, such methods only consist of local solvers complemented by a crude global step. The method introduced below 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. Numerical observations confirm the major practical interest methodological improvement.

Why is a *linear scaling* plausible for computing $D_\star$? To justify the fact that the cubic scaling is an estimate by excess of the computational task required to solve (1.1), we argue that the matrix does not need to be diagonalized. As mentioned above, only the *orthogonal projector* on the subspace generated by the lowest N eigenvectors is to be determined and *not* the *explicit* values of these lowest N eigenvectors. But in order to reach a linear complexity, appropriate assumptions are necessary, both on the form of the matrices H and S, and on the matrix $D_\star$ solution to (1.1):

- (H1). The matrices H and S are assumed 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 follows from (1.3) and (1.4) that it is automatically satisfied for Kohn–Sham models as soon as the basis functions are localized in real space, which is in particular the case for the widely used atomic orbital basis sets [7].
- (H2). A second assumption is that the matrix $D_\star$ built from the solution to (1.1) is also sparse. This condition seems to be fulfilled as soon as the relative gap

$$\gamma = \frac{\epsilon_{N+1} - \epsilon_N}{\epsilon_{N_b} - \epsilon_1} \tag{1.7}$$

deduced from the solution of (1.1) is large enough. As explained in Section 2 below, this observation can be supported by qualitative physical arguments. On the other hand, we are not aware of any mathematical argument of linear algebra that would justify assumption (H2) in a general setting.

We assume (H1)–(H2) in the following. Current efforts aim at treating cases when the second assumption is not fulfilled, which in particular corresponds to the case of conducting materials. The problem (1.2) is then extremely difficult because the gap γ in (1.7) being very small, the matrix D is likely to be dense. Reaching linear complexity is then a challenging issue, subject of ongoing works by several groups of researchers [8–13]. It

therefore makes sense to improve in a first step the existing methods in the setting of assumption (H2), before turning to more challenging issues.

Before we get to the heart of the matter, we would like to point out the following feature of the problem under consideration.

In practice, problem (1.1) has to be solved *repeatedly*. For instance, it is the inner loop in a nonlinear minimization problem where H depends self-consistently on $D_\star$. We refer to [14,20] for efficient algorithms to iterate on this nonlinearity and to [7] for a review on the subject. Alternatively, or in addition to the above, problem (1.1) is parameterized by the positions of the nuclei (both the mean-field operator H and the overlap matrix S indeed depend on these positions), and these positions may vary. This is the case in molecular mechanics (find the optimal configuration of nuclei that gives the lowest possible energy to the molecular system), and in molecular dynamics as well (the positions of nuclei follow the Newton law of motion in the mean-field created by the electrons). In either case, problem (1.1) is not be solved *from scratch*. Because of previous calculations, we may consider we have at our disposal a good initial guess for the solution. The latter comes from e.g. previous positions of nuclei, or previous iterations in the outer loop of determination of H. In difficult cases it may even come from a previous computation with a coarse grained model. In other words, the question addressed reads *solving Problem (1.1) for some* $H + \delta H$ *and* $S + \delta S$ *that are small perturbations of previous* H *and* S *for which the solution is known*. This specific context allows for a speed up of the algorithm when the initial guess is sufficiently good. This is the reason why, in the following, we shall frequently make distinctions between bad and good initial guesses.

2. Localization in quantum chemistry

The physical system we consider is a long linear molecule (for instance a one-dimensional polymer or a nanotube). Let us emphasize that we do not claim a particular physical relevance of this system. This is for the purpose of illustration. We believe the system considered to be a good representative of a broad class of large molecular systems that may be encountered practically. Considering a one-dimensional situation is of course an oversimplification of all the technicalities of the general three-dimensional problem that is of practical interest. However, from the mathematical viewpoint, it does not significantly change the landscape. This is for the purpose of avoiding unnecessary technicalities related to the "geometrical" tiling of the molecular system into a set of disjoint zones. Section 3.5 below describes the necessary adaptation of our strategy to allow for the treatment of three-dimensional systems.

Our one-dimensional setting is the following. Each atomic orbital χ_i is centered on one nucleus. Either it is supported in a ball of small radius [26] (in comparison to the size of the macromolecule under study), or it has a rapid exponential-like or Gaussian-like [16] fall-off. The atomic orbitals are numbered following the orientation of the molecule. Then, the mean-field Hamiltonian matrix H whose entries are defined by (1.3) has the band structure shown in Fig. 1.

Although the eigenvectors of H are *a priori* delocalized (most of their coefficients do not vanish), it seems to be possible to build a S-orthonormal basis of the subspace generated by the lowest N eigenvectors of H, consisting of *localized* vectors (only a few consecutive coefficients are non-zero). This is motivated by a physical argument of locality of the interactions [22]. For periodic systems, the localized vectors correspond to the so-called Wannier orbitals [4]. It can be proven that in this case, the larger the band gap, the better the localiza-

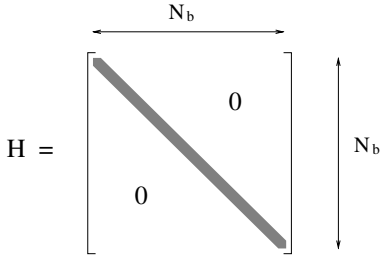


Fig. 1. Band structure of the symmetric matrix H.

tion of the Wannier orbitals [21]. For insulators, the Wannier orbitals indeed enjoy an exponential fall-off rate proportional to the band gap. For conductors, the fall-off is only algebraic. As mentioned in Section 1, we only consider here the former case. This allows us to assume that there exists some integer $q \ll N_b$, such that N_b/q is an integer, for which all of these localized functions can be essentially expanded on q consecutive atomic orbitals. Denoting by $n = 2q$, we can therefore assume a good approximation of a solution $C_\star$ to (1.6) exists, with the block structure displayed in Fig. 2. Note that each block C_i only overlaps with its nearest neighbors. Correspondingly, we introduce the block structure of H displayed in Fig. 3. The matrix D constructed from a block matrix C using (1.5) has the structure represented in Fig. 4 and satisfies the constraints $D = D^t$, $D^2 = D$, $\mathrm{Tr}(D) = N$.

Let us point out that the integers q and $n = 2q$ depend on the band gap, *not* on the size of the molecule. The condition $n = 2q$ is only valid for $S = I_{N_b}$. For $S \neq I_{N_b}$, it is replaced by $n = 2q + nbs$ where $2nbs - 1$ is the bandwidth of the matrix S.

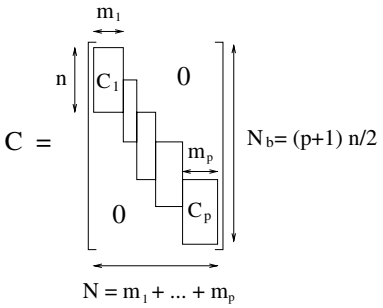


Fig. 2. Block structure of the matrices C. Note that by construction each block only overlaps with its nearest neighbors.

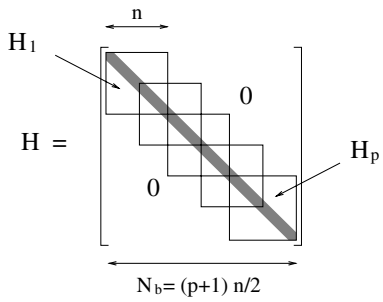


Fig. 3. Block structure of the matrix H.

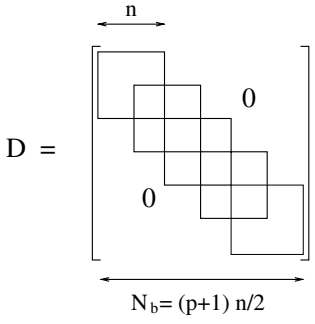


Fig. 4. Block structure of the matrix D.

The domain decomposition algorithm we propose aims at searching an approximate solution to (1.6) that has the block structure described above.

For simplicity, we now present our method assuming that $S = I_{N_b}$, i.e. that the Galerkin basis $\{\chi_i\}_{1\leqslant i\leqslant N_b}$ is orthonormal. The extension of the method to the case when $S \neq I_{N_b}$ is straightforward. Problem (1.6) then reads

$$\inf\left\{\mathrm{Tr}(HCC^t),\quad C \in \mathscr{M}^{N_b,N}(\mathbb{R}),\ C^tC = I_N\right\}. \tag{2.1}$$

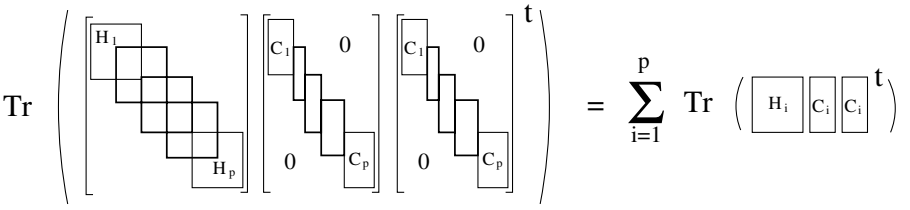
Our approach consists in solving an approximation of problem (2.1) obtained by minimizing the exact energy $\mathrm{Tr}(HCC^t)$ on the set of the matrices C which have the block structure displayed in Fig. 2 and satisfy the constraint $C^tC = I_N$. The resulting minimization problem can be recast as

$$\inf\left\{\sum_{i=1}^{p}\mathrm{Tr}(H_iC_iC_i^t),\qquad C_i \in \mathscr{M}^{n,m_i}(\mathbb{R}),\ m_i \in \mathbb{N},\quad C_i^tC_i = I_{m_i}\ \forall 1\leqslant i\leqslant p,\right.$$
$$\left. C_i^tTC_{i+1} = 0\ \forall 1\leqslant i\leqslant p-1,\quad \sum_{i=1}^{p} m_i = N\right\}. \tag{2.2}$$

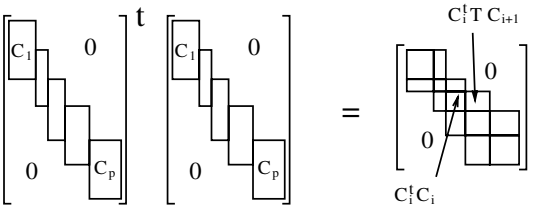
In the above formula, $T \in \mathscr{M}^{n,n}(\mathbb{R})$ is the matrix defined by

$$T_{kl} = \begin{cases} 1 & \text{if } k-l = q,\\ 0 & \text{otherwise,}\end{cases} \tag{2.3}$$

and $H_i \in \mathscr{M}^{n,n}(\mathbb{R})$ is a symmetric submatrix of H (see Fig. 3). 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 $\sum_{i=1}^{p-1} m_im_{i+1}$ *local* scalar constraints $C_i^tTC_{i+1} = 0$, involving vectors of size n. We would like to emphasize that we can obtain in this way a basis of the vector space generated by the lowest N eigenvectors of H, but not the eigenvectors themselves. This method is therefore not directly applicable to standard diagonalization problems.

Our algorithm searches for the solution to (2.2), not to (2.1). More rigorously stated, we search for the solution to the Euler–Lagrange equations of (2.2):

$$\begin{cases} H_iC_i = C_iE_i + T^tC_{i-1}\Lambda_{i-1,i} + TC_{i+1}\Lambda_{i,i+1}^t, & 1\leqslant i\leqslant p,\\ C_i^tC_i = I_{m_i}, & 1\leqslant i\leqslant p,\\ C_i^tTC_{i+1} = 0, & 1\leqslant i\leqslant p-1,\end{cases} \tag{2.4}$$

where by convention

$$C_0 = C_{p+1} = 0. \tag{2.5}$$

The matrices $(E_i)_{1\leqslant i\leqslant p}$ and $(\Lambda_{i,i+1})_{1\leqslant i\leqslant p-1}$, respectively, denote the matrices of Lagrange multipliers associated with the orthonormality constraints $C_i^t C_i = I_{m_i}$ and $C_i^t T C_{i+1} = 0$. The $m_i \times m_i$ matrix E_i is symmetric. The matrix $\Lambda_{i,i+1}$ is of size $m_i \times m_{i+1}$. The above equations can be easily derived by considering the Lagrangian

$$\mathscr{L}(\{C_i\},\{E_i\},\{\Lambda_{i,i+1}\}) = \sum_{i=1}^{p} \mathrm{Tr}(H_i C_i C_i^t) + \sum\nolimits_{i=1}^{p} \mathrm{Tr}((C_i^t C_i - I_{m_i})E_i) + \sum_{i=1}^{p-1} \mathrm{Tr}(C_i^t T C_{i+1} \Lambda_{i,i+1}^t).$$

The block structure imposed on the matrices clearly lowers the dimension of the search space we have to explore. However, this simplification comes at a price. First, problem (2.2) only *approximates* problem (2.1). Second, (2.2) may have local, non-global, minimizers, whereas all the local minimizers of (2.1) are global. There are thus *a priori* many spurious solutions of the Euler Lagrange equations (2.4) associated with (2.2).

A point is that the sizes $(m_i)_{1\leqslant i\leqslant p}$ are not *a priori* prescribed. In our approach, they are adjusted during the iterations. We shall see how in the sequel.

3. Description of the domain decomposition algorithm

3.1. Description of a simplified form

For pedagogic purpose, we first consider the following problem

$$\inf\left\{\langle H_1 Z_1, Z_1\rangle + \langle H_2 Z_2, Z_2\rangle, \quad Z_i \in \mathbb{R}^{N_b},\ \langle Z_i, Z_i\rangle = 1,\ \langle Z_1, Z_2\rangle = 0\right\}. \tag{3.1}$$

Problem (3.1) is a particular occurrence of (2.2). We have denoted by $\langle\cdot,\cdot\rangle$ the standard Euclidean scalar product on $\mathbb{R}^{N_b}$.

For (3.1), the algorithm is defined in the following simplified form. 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

$$\begin{cases} \widetilde{Z}_1^k = \mathrm{arginf}\{\langle H_1 Z_1, Z_1\rangle, \quad Z_1 \in \mathbb{R}^{N_b},\ \langle Z_1, Z_1\rangle = 1,\ \langle Z_1, Z_2^k\rangle = 0\}, \\ \widetilde{Z}_2^k = \mathrm{arginf}\{\langle H_2 Z_2, Z_2\rangle, \quad Z_2 \in \mathbb{R}^{N_b},\ \langle Z_2, Z_2\rangle = 1,\ \langle \widetilde{Z}_1^k, Z_2\rangle = 0\}. \end{cases} \tag{3.2}$$

- *Global step:* Solve

$$\alpha^* = \mathrm{arginf}\{\langle H_1 Z_1, Z_1\rangle + \langle H_2 Z_2, Z_2\rangle, \quad \alpha \in \mathbb{R}\}, \tag{3.3}$$

where

$$Z_1 = \frac{\widetilde{Z}_1^k + \alpha \widetilde{Z}_2^k}{\sqrt{1+\alpha^2}}, \quad Z_2 = \frac{-\alpha \widetilde{Z}_1^k + \widetilde{Z}_2^k}{\sqrt{1+\alpha^2}} \tag{3.4}$$

and set

$$Z_1^{k+1} = \frac{\widetilde{Z}_1^k + \alpha^* \widetilde{Z}_2^k}{\sqrt{1+(\alpha^*)^2}}, \quad Z_2^{k+1} = \frac{-\alpha^* \widetilde{Z}_1^k + \widetilde{Z}_2^k}{\sqrt{1+(\alpha^*)^2}}. \tag{3.5}$$

In the kth iteration of the local step, we first fix $Z_2 = Z_2^k$ and optimize over Z_1 to obtain $\widetilde{Z}_1^k$. Then we fix $Z_1 = \widetilde{Z}_1^k$ and optimize over Z_2 to obtain $\widetilde{Z}_2^k$. This 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. In the global step, we optimize the *sum* $\langle H_1 Z_1, Z_1\rangle + \langle H_2 Z_2, Z_2\rangle$ over the subspace spanned by $\widetilde{Z}_1^k$ and $\widetilde{Z}_2^k$, subject to the constraints in (3.1). 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. It can be shown that the combined algorithm (local step + global step) monotonically decreases the objective function and globally converges to an optimal solution of (3.1).

This algorithm operates at two levels: a fine level where we solve two problems of dimension N_b rather than one problem of dimension $2N_b$; a coarse level where we solve a problem of dimension 2. Left by itself, the fine step converges to a suboptimal solution of (3.1). Combining the fine step with the global step yields convergence to a global optimum.

In addition to providing a pedagogic view on the general algorithm presented in the following section, the simplified form (3.2)–(3.5) has a theoretical interest. In contrast to the general algorithm for which we cannot provide a convergence analysis, the simplified form (3.2)–(3.5) may be analyzed mathematically, at least in the particular situation when $H_1 = H_2 = H$. Then solving (3.1) amounts to searching for the lowest two eigenelements of the matrix H. Notice that the global step (3.3)–(3.5) is then unnecessary because the functional to minimize in (3.3) does not depend on α.

However, we can show that the iterations (3.2) converge in the following sense. The two-dimensional vector space spanned by the lowest two eigenvalues of H is reached asymptotically. This occurs under an appropriate condition on the matrix H. The latter is a condition of separation of the eigenvalues, namely $\epsilon_2 - \epsilon_1 < \epsilon_3 - \epsilon_2$ with obvious notation. The gap $\epsilon_3 - \epsilon_2$ gives the speed of convergence. For brevity, we do not detail the proof here (see [5]). Future work on the numerical analysis of more general cases is in progress.

3.2. Description of the algorithm

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

$$\mathscr{E}((C_i)_{1\leqslant i\leqslant p}) = \sum_{i=1}^{p} \mathrm{Tr}(H_i C_i C_i^t), \tag{3.6}$$

and set by convention

$$U_0 = U_p = 0. \tag{3.7}$$

We introduce an integer ϵ, initialized to one, that will alternate between the values zero and one during the iterations.

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

3.2.1. Multilevel domain decomposition algorithm

- **Step 1: Local fine solver.**
  - (a) For each i, diagonalize the matrix $H_{2i+\epsilon}$ in the subspace

$$V_{2i+\epsilon}^k = \left\{ x \in \mathbb{R}^n, \quad \left[C_{2i+\epsilon-1}^k\right]^t T x = 0, \quad x^t T C_{2i+\epsilon+1}^k = 0 \right\},$$

i.e. diagonalize $P_{2i+\epsilon}^k H_{2i+\epsilon} P_{2i+\epsilon}^k$ where $P_{2i+\epsilon}^k$ is the orthogonal projector on $V_{2i+\epsilon}^k$. This provides (at least) $n - m_{2i+\epsilon-1}^k - m_{2i+\epsilon+1}^k$ real eigenvalues $\lambda_{2i+\epsilon,1}^k \leqslant \lambda_{2i+\epsilon,2}^k \leqslant \cdots$ and associated orthonormal vectors $x_{2i+\epsilon,j}^k$. The latter are T-orthogonal to the column vectors of C_{i-1}^k and C_{i+1}^k.

  - (b) Sort the eigenvalues $(\lambda_{2i+\epsilon,j}^k)_{i,j}$ in increasing order, and select the lowest $\sum_i m_{2i+\epsilon}$ of them. For each i, collect in block $\#2i+\epsilon$ the eigenvalues $\lambda_{2i+\epsilon,j}^k$ selected. New intermediate block sizes $\bar{m}_{2i+\epsilon}^k$ are defined.
  - (c) For each i, collect the lowest $\bar{m}_{2i+\epsilon}^k$ vectors $x_{2i+\epsilon,j}^k$ in the $n \times \bar{m}_{2i+\epsilon}^k$ matrix $\overline{C}_{2i+\epsilon}^k$.
  - (d) For each i, diagonalize the matrix $H_{2i+\epsilon+1}$ in the subspace

$$V_{2i+\epsilon+1}^k = \left\{ x \in \mathbb{R}^n, \quad \left[\overline{C}_{2i+\epsilon}^k\right]^t T x = 0, \quad x^t T \overline{C}_{2i+\epsilon+2}^k = 0 \right\}$$

in order to get eigenvalues $\lambda_{2i+\epsilon+1,1}^k \leqslant \lambda_{2i+\epsilon+1,2}^k \leqslant \cdots$ and associated orthonormal vectors $x_{2i+\epsilon+1,j}^k$. The latter are T-orthogonal to the column vectors of $\overline{C}_{2i+\epsilon}^k$ and $\overline{C}_{2i+\epsilon+2}^k$.

(e) Sort all the eigenvalues $\{(\lambda^k_{2i+\epsilon+1,j})_{i,j}, (\lambda^k_{2i+\epsilon,j})_{i,j}\}$ in increasing order. Select the lowest N. For each l, collect in block #l the eigenvalues $\lambda^k_{l,j}$ selected. New intermediate block sizes $(m_l^{k+1})_{1\leqslant l\leqslant p}$ are thus defined.
(f) Set $\widetilde{C}^k_l = [x^k_{l,1}|\cdots|x^k_{l,m_l^{k+1}}]$.
(g) Replace ϵ by $1-\epsilon$ and proceed to step 2 below.

- **Step 2: global coarse solver.** Solve

$$\mathscr{U}^* = \operatorname{arginf}\{f(\mathscr{U}), \quad \mathscr{U} = (U_i)_i \ \forall 1 \leqslant i \leqslant p-1,\ U_i \in \mathscr{M}^{m_{i+1},m_i}(\mathbb{R})\}, \tag{3.8}$$

where

$$f(\mathscr{U}) = \mathscr{E}\left(\left(C_i(\mathscr{U})(C_i(\mathscr{U})^t C_i(\mathscr{U}))^{-1/2}\right)_i\right), \tag{3.9}$$

and

$$C_i(\mathscr{U}) = \widetilde{C}^k_i + T\widetilde{C}^k_{i+1}U_i([\widetilde{C}^k_i]^t TT^t\widetilde{C}^k_i) - T^t\widetilde{C}^k_{i-1}U^t_{i-1}([\widetilde{C}^k_i]^t T^t T\widetilde{C}^k_i). \tag{3.10}$$

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

$$C^{k+1}_i = C_i(\mathscr{U}^*)(C_i(\mathscr{U}^*)^t C_i(\mathscr{U}^*))^{-1/2}. \tag{3.11}$$

Note that $[C^{k+1}_i]^t TC^{k+1}_{i+1} = 0$ (this follows from $T^2 = 0$).

We think of the even indexed unknowns C_{2i} as the black variables and the odd indexed unknowns C_{2i+1} as the white variables. In the first phase of the local fine solver, we optimize over the white variables while holding the black variables fixed. In the second phase of the local fine solver, we optimize over the black variables while holding the white variables fixed. In the global step, we perturb each variable by a linear combination of the adjacent variables. The matrices $\mathscr{U} = (U_i)_i$ in (3.8) play the same role as the real parameter α in (3.3). The perturbation is designed so that the constraints are satisfied. The optimization is performed over the matrices generating the linear combinations. In the next iteration, we interchange the order of the optimizations: first optimize over the black variables while holding the white variables fixed, then optimize over the white variables while holding the black variables fixed.

Let us point out that an accurate solution to (3.8) is not needed. In practice, we reduce the computational cost of the global step, by using again a domain decomposition method. The blocks $(C_i)_{1\leqslant i\leqslant p}$ are collected in r overlapping groups $(G_l)_{1\leqslant l\leqslant r}$ as shown in Fig. 5. Problem (3.8) is solved first for the blocks (G_{2l+1}), next for the blocks (G_{2l}). Possibly, this procedure is repeated a few times. The advantage of this strategy is that the computational time of the global step scales linearly with N. In addition, it is parallel in nature. The solution of (3.8) for a given group is performed by a few steps of a Newton-type algorithm. Other preconditioned iterative methods could also be considered.

3.3. Comments on the local step

The local step is based on a checkerboard iteration technique.
When $\epsilon = 1$, steps 1a–1c search for a solution $(\bar{m}^k_{2i+1}, \overline{C}^k_{2i+1})_i$ to the problem

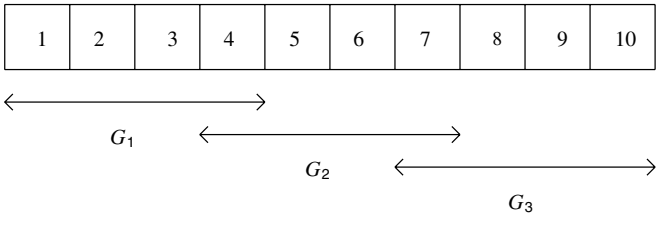


Fig. 5. Collection of $p = 10$ blocks into $r = 3$ groups.

$$\inf\left\{\sum_i \mathrm{Tr}\big(H_{2i+1}C_{2i+1}C_{2i+1}^t\big),\qquad C_{2i+1}\in\mathscr{M}^{n,m_{2i+1}}(\mathbb{R}),\ C_{2i+1}^tC_{2i+1}=I_{m_{2i+1}},\right.$$
$$\left.[C_{2i}^k]^tTC_{2i+1}=0,\ C_{2i+1}^tTC_{2i+2}^k=0,\quad m_{2i+1}\in\mathbb{N},\ \sum_i m_{2i+1}=\sum_i m_{2i+1}^k\right\}.$$

During steps 1a–1c, the "white" blocks C_{2i}^k are kept fixed. The "black" blocks C_{2i+1}^k are optimized under the orthogonality constraints imposed by the "white" blocks. A point is that most of the computational effort can be done *in parallel*. Indeed, for p even, say, performing step 1a amounts to solving $p/2$ *independent* diagonalization problems of size n.

Likewise, steps 1d–1f solve

$$\inf\left\{\sum_{i=1}^p \mathrm{Tr}\big(H_iC_iC_i^t\big),\qquad C_i\in\mathscr{M}^{n,m_i}(\mathbb{R}),\ C_i^tC_i=I_{m_i},\ m_i\in\mathbb{N},\ \sum_i m_i=N,\right.$$
$$\left.[\overline{C}_{2j-1}^k]^tTC_{2j}=0,\ [C_{2j}]^tT[\overline{C}_{2j+1}]^k=0,\quad 0\leqslant m_{2j+1}\leqslant \bar{m}_{2j+1}^k,\ C_{2j+1}\subset\overline{C}_{2j+1}^k\right\},$$

where the notation $C_{2j+1}\subset\overline{C}_{2j+1}^k$ means that each column of C_{2j+1} is a column of $\overline{C}_{2j+1}^k$. Here again, most of the computational effort can be performed in parallel.

When $\epsilon=1$, "black" vectors (i.e. vectors belonging to blocks with odd indices) are allowed to become "white" vectors, but the reverse is forbidden. In order to symmetrize the process, ϵ is replaced by $1-\epsilon$ in the next iteration.

We wish to emphasize that, although called *local*, this step already accounts for some global concern. Indeed, and it is a key point of the local step, substeps (b) and (e) sort the *complete* set of eigenvalues generated locally. This, together with the update of the size m_i of the blocks, allows for a preliminary propagation of the information throughout the whole system. The global step will complement this.

Finally, let us mention that in the local steps, (approximate) T-orthogonality is obtained by a Householder orthonormalization process. The required orthonormality criterion is

$$\forall 1\leqslant i\leqslant p-1,\quad \left\|[\widetilde{C}_i^k]^tT\widetilde{C}_{i+1}^k\right\|\leqslant\epsilon_L, \tag{3.12}$$

where $\epsilon_L>0$ is a threshold to be chosen by the user.

3.4. Comments on the global step

Let us briefly illustrate the role played by the global step. For simplicity, we consider the case of two blocks of same initial size $m_1=m_2=m$ and we assume that m_1 and m_2 do not vary during the iterations. If only the local step is performed, then the new iterate

$$(C_1^{k+1},C_2^{k+1})=(\widetilde{C}_1^k,\widetilde{C}_2^k)$$

does not necessarily satisfies (2.4). Indeed, there is no reason why the Lagrange multipliers corresponding to the two constraints $C^tTC_2^k=0$ (step 1a when $\epsilon=1$) on the one hand and $[\widetilde{C}_1^k]^tTC=0$ (step 1d when $\epsilon=1$) on the other hand should be the same. The global step *asymptotically* enforces the equality of Lagrange multipliers. This is a way to account for a global feature of the problem.

Let us emphasize this specific point. Assume $U^*=0$ in the global step of the kth iteration of the algorithm, or in other words that the global step is not effective at the kth iteration. Then it implies that the output $(\widetilde{C}_1,\widetilde{C}_2)=(\widetilde{C}_1^k,\widetilde{C}_2^k)$ of the local step already satisfies (2.4). Indeed,

$$f(U)=\mathrm{Tr}(J_1(U)C_1(U)^tH_1C_1(U))+\mathrm{Tr}(J_2(U)C_2(U)^tH_2C_2(U)) \tag{3.13}$$

with $J_i(U)=(C_i(U)^tC_i(U))^{-1}$ for $i=1,2$. Since

$$(J_1(U))^{-1} = I_m + \left(\widetilde{C}_1^t T T^t \widetilde{C}_1\right) U^t \left(\widetilde{C}_2^t T^t T \widetilde{C}_2\right) U \left(\widetilde{C}_1^t T T^t \widetilde{C}_1\right),$$
$$(J_2(U))^{-1} = I_m + \left(\widetilde{C}_2^t T^t T \widetilde{C}_2\right) U \left(\widetilde{C}_1^t T T^t \widetilde{C}_1\right) U^t \left(\widetilde{C}_2^t T^t T \widetilde{C}_2\right),$$

we have $\nabla J_1(0) = \nabla J_2(0) = 0$. The matrix U being a square matrix of dimension m, for all $1 \leqslant i, j \leqslant m$,

$$\begin{aligned}\frac{1}{2}\frac{\partial f}{\partial U_{ij}}(0) &= \mathrm{Tr}\left(\left[\frac{\partial C_1}{\partial U_{ij}}(0)\right]^t H_1 \widetilde{C}_1\right) + \mathrm{Tr}\left(\left[\frac{\partial C_2}{\partial U_{ij}}(0)\right]^t H_2 \widetilde{C}_2\right)\\ &= \left(\left(\widetilde{C}_1^t T T^t \widetilde{C}_1\right)\widetilde{C}_1^t H_1 T \widetilde{C}_2\right)_{ji} - \left(\widetilde{C}_1^t T H_2 \widetilde{C}_2 \left(\widetilde{C}_2^t T^t T \widetilde{C}_2\right)\right)_{ji}\\ &= \left(\left(\widetilde{C}_1^t T T^t \widetilde{C}_1\right)(\Lambda_1 - \Lambda_2)\left(\widetilde{C}_2^t T^t T \widetilde{C}_2\right)\right)_{ji}, \end{aligned} \tag{3.14}$$

where Λ_1 and Λ_2 are defined by

$$\begin{cases} H_1 \widetilde{C}_1 = \widetilde{C}_1 E_1 + T \widetilde{C}_2 \Lambda_1^t, \\ H_2 \widetilde{C}_2 = \widetilde{C}_2 E_2 + T^t \widetilde{C}_1 \Lambda_2. \end{cases} \tag{3.15}$$

As $U^* = 0$ implies

$$\forall 1 \leqslant i, j \leqslant m, \quad \frac{\partial f}{\partial U_{ij}}(0) = 0, \tag{3.16}$$

we conclude that $\Lambda_1 = \Lambda_2$ if the matrices $\left(\widetilde{C}_1^t T T^t \widetilde{C}_1\right)$ and $\left(\widetilde{C}_2^t T^t T \widetilde{C}_2\right)$ are invertible, which is generally the case when $n \gg 2m$. Consequently, (2.4) is satisfied by $(\widetilde{C}_1, \widetilde{C}_2)$.

On the other hand, when n is not much larger that $2m$, the above matrices are not invertible and (2.4) is usually not satisfied. In this case, the global step is slightly modified in order to recover (2.4) and thus improve the efficiency of the global step. We replace (3.10) by

$$\forall 1 \leqslant i \leqslant p, \quad C_i(\mathscr{U}) = \widetilde{C}_i^k + T \widehat{C}_{i+1}^k U_i \left([\widehat{C}_i^k]^t T T^t \widehat{C}_i^k\right) - T^t \widehat{C}_{i-1}^k U_{i-1}^t \left([\widehat{C}_i^k]^t T^t T \widehat{C}_i^k\right), \tag{3.17}$$

where $\widehat{C}_i^k$ is a block formed by vectors collected in the vector space defined by $\widetilde{C}_i^k$. These vectors are selected using a modified Gram–Schmidt orthonormalization process. The size of the blocks $\widehat{C}_i^k$ is appropriately chosen. The larger the blocks $\widehat{C}_i^k$, the more precise the global step but the worse the conditioning of the optimization problem. In addition, since the global step is the most demanding step of the algorithm, considerations both on the computational time and in terms of memory are accounted for when fixing the sizes of the blocks $\widehat{C}_i^k$.

Our numerical experiments show that when the global step is performed (using (3.10) or (3.17), depending on n and m), the blocks $(C_i^{k+1})_i$ do not exactly satisfy the orthonormality constraint, owing to evident round-off errors. All the linear scaling algorithms have difficulties in ensuring this constraint and our MDD approach is no exception. The tests performed however show that the constraint remains satisfied throughout the iterations within a good degree of accuracy.

3.5. On systems of higher dimensions

The present section briefly describes how the above strategy should be adapted to treat three-dimensional systems.

In the one-dimensional situation addressed above that is, when the physical system under study is, say, a linear polymer, the system is decomposed into a sequence of segments. Each segment overlaps with the previous and the next ones, in the sense that the basis functions attached to this segment have a non-zero overlap with the basis functions of the adjacent segments. On the other hand, in a system of dimension higher than or equal to two, the number of adjacent zones increases. In dimension 1, considering the overlap of the nth block with the $(n-1)$th one and the $(n+1)$th one is enough. In dimension 2, say for a thin film, there are 8 such neighboring blocks, in dimension 3 (case of a bulk crystal) there are 26. All this is a technical issue, which will definitely have an impact on the difficulty of the implementation of the method. It will have only a limited

impact on its efficiency. In a nutshell, the "black and white" parallel strategy described in the previous sections needs to be adequately modified. But this is a generic difficulty of all domain decomposition method in the engineering sciences, which is thus well documented and controlled. Additionally, an increasing number of orthogonality constraints is to be accounted for in the local problems, but, again, this is no conceptual obstruction. Current research is directed toward the adaptation of our method to the multidimensional case, following the lines described above.

4. Numerical tests

An extensive set of numerical tests was performed to illustrate the important features of the domain decomposition algorithm introduced above, and to compare it with a standard scheme, commonly used in large scale electronic structure calculations.

4.1. Setting of the algorithm and of the tests

4.1.1. Molecular systems used for the tests

Numerical tests on the algorithm presented above were performed on three chemical systems. The first two systems both have formula $COH–(CO)_{n_m}–COH$. They differ in their carbon–carbon interatomic distances. For system $\mathscr{P}_1$, this distance is fixed to 5 atomic units, while it is fixed to 4 for system $\mathscr{P}_2$. On the other hand, our third system, denoted by $\mathscr{P}_3$ has formula $CH_3–(CH_2)_{n_m}–CH_3$.

For each of the three systems $\mathscr{P}_1$, $\mathscr{P}_2$, $\mathscr{P}_3$, several numbers n_m of monomers were considered. A geometry optimization was performed using the GAUSSIAN package [32] in order to fix the internal geometrical parameters of the system. The only exception to this is the carbon–carbon distance for $\mathscr{P}_1$ and $\mathscr{P}_2$, which, as said above, is fixed *a priori*. Imposing the carbon–carbon distance allows to control the sparsity of the matrices H and S (the larger the distance, the sparser the matrices). Although not physically relevant, fixing the carbon–carbon distance is therefore useful for the purpose of numerical tests.

4.1.2. Data, parameters and initialization

For an extremely large number n_m of monomers, the matrices H, S, and $D_\star$ cannot be generated directly with the GAUSSIAN package. We therefore make a periodicity assumption. For large values of n_m, these matrices approach a periodic pattern (leaving apart, of course, the "boundary layer", that is the terms involving orbitals close to one end of the linear molecule). So, we first fix some n_m sufficiently large, but for which a direct calculation with Gaussian is feasible, and construct H, S. The matrices H and S, as well as the ground-state density matrix $D_\star$, and the ground-state energy E_0, are then obtained for arbitrary large n_m assuming periodicity out of the "boundary layer". Likewise, the gap γ in the eigenvalues of H is observed to be constant, for each system, irrespective of the number n_m of polymers, supposedly large. Proceeding so, the gap for systems $\mathscr{P}_1$, $\mathscr{P}_2$, and $\mathscr{P}_3$ is, respectively, evaluated to 0.00104, 0.00357, and 0.0281.

For our MDD approach, localization parameters are needed. They are shown in Table 1 below. Additionally, we need to provide the algorithm with an initial guess on the size m_i of the blocks. Based on physical

Table 1
Localization parameters and initial size of the blocks used in the tests

	$\mathscr{P}_1$	$\mathscr{P}_2$	$\mathscr{P}_3$
n	130	200	308
q	50	80	126
Bandwidth of S	59	79	111
Bandwidth of H	99	159	255
Cut-off for entries of H	10^{-12}	10^{-12}	10^{-10}
Cut-off for entries of D	10^{-11}	10^{-11}	10^{-7}
Size of first block	$m_1 = 67$	$m_1 = 105$	$m_1 = 136$
Size of last block	$m_p = 67$	$m_p = 106$	$m_p = 137$
Size of a generic block	$m_i = 56$	$m_i = 84$	$m_i = 104$

considerations on the expected repartition of the electrons in the molecule and on the expected localization of the orbitals, the sizes were fixed to values indicated in Table 1. The specific block C_i is then initialized in one of the following three manners:

- strategy $\mathscr{I}_1$: the entries of C are generated randomly, which of course generically yields a bad initial guess way;
- strategy $\mathscr{I}_2$: each block C_i consists of the lowest m_i (generalized) eigenvectors associated to the corresponding block matrices H_i and S_i in the matrices H and S, respectively. This provides with an initial guess, depending on the matrices H and S, thus of better quality than the random one provided by strategy $\mathscr{I}_1$;
- strategy $\mathscr{I}_3$: the initial guess provided by $\mathscr{I}_2$ is optimized with the local fine solver described in Section 3.2.

4.1.3. Implementation details

Exact diagonalizations in the local steps are performed with the routine *dsbgv.f* from the LAPACK package [1]. In the global step, the resolution of the linear system involving the Hessian matrix is performed iteratively, using SYMMLQ [31]. Diagonal preconditioning is used to speed up the resolution.

The calculations have been performed using only one processor of a bi-processor Intel Pentium IV-2.8 GHz.

4.1.4. Criteria for comparison of results

For assessment of the quality of the results, we have used two criteria, regarding the ground-state energy and the ground-state density matrix, respectively. For either quantity, the reference calculation is the calculation using the Gaussian package [32]. The quality of the energy is measured using the relative error $e_E = \frac{|E-E_0|}{|E_0|}$. For evaluation of the quality of the density matrix, we use the L^∞ matrix norm

$$e_\infty = \sup_{(i,j) \text{ s.t. } |H_{ij}| \leqslant \varepsilon} |D_{ij} - [D_\star]_{ij}|, \tag{3.18}$$

where we fix $\varepsilon = 10^{-10}$. The introduction of the norm (3.18) is consistent with the cut-off performed on the entries of H (thus the exact value of ε chosen). Indeed, in practice, the matrix D is only used for the calculations of various observables (for instance electronic energy and Hellman–Feynman forces), all of the form $\mathrm{Tr}(AD)$ where the symmetric matrix A shares the same pattern as the matrix H (see [7] for details). The result is therefore not sensitive to entries with indices (i,j) such that $|H_{ij}|$ is below the cut-off value.

4.2. Illustration of the role of the local and global steps

Our MDD method consists in three ingredients:

- the local optimization of each block performed in the local step;
- the transfer of vectors from some blocks to other blocks, along with the modification of the block sizes m_i, again in the local step;
- the optimization performed in the global step.

To highlight the necessity of each of the ingredients, and their impacts on the final result, we compare our MDD algorithm with three simplified variants. Let us denote by:

- strategy $\mathscr{S}_1$: local optimization of the blocks, without allowing variations of the block sizes, and no global step;
- strategy $\mathscr{S}_2$: full local step (as defined in Section 3.2), no global step;
- strategy $\mathscr{S}_3$: local optimization of the blocks, without allowing for variations of the block sizes, and global step;
- strategy $\mathscr{S}_4$: full algorithm.

We compare the rate of convergence for the above four strategies. Two categories of tests are performed, depending on the quality of the initial guess. The results displayed in Figs. 6–9 concern polymer $\mathscr{P}_1$ with $n_m = 801$ monomers. This corresponds to $N_b = 8050$ and $N = 5622$. Analogous tests were performed on $\mathscr{P}_2$ and $\mathscr{P}_3$, but we do not present them here, for brevity.

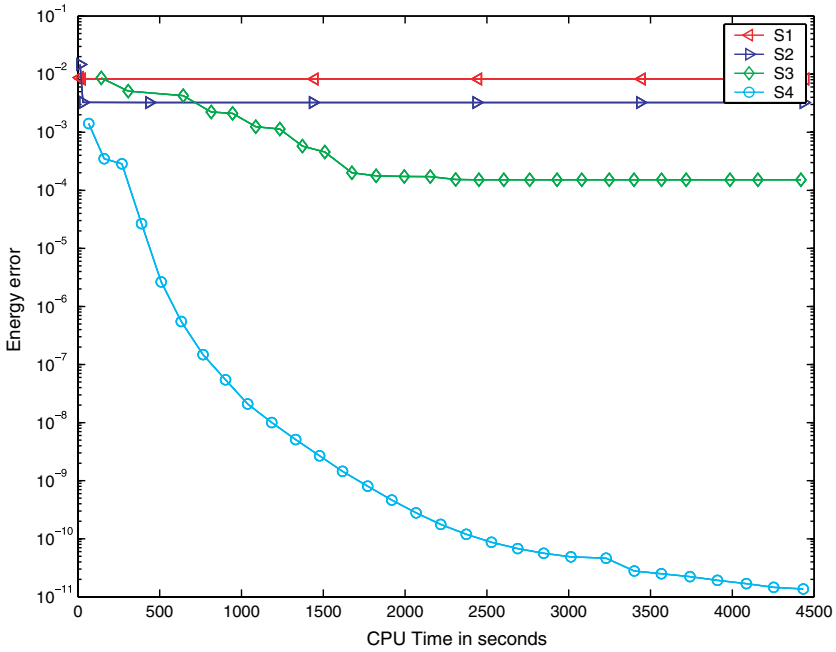


Fig. 6. Energy error versus CPU time obtained with a bad initial guess ($\mathscr{I}_1$).

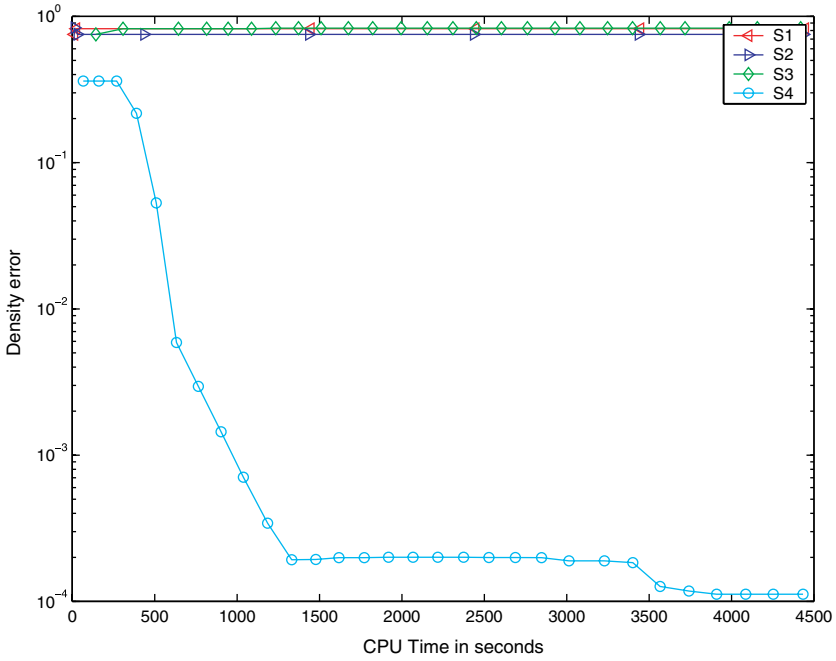


Fig. 7. Density error versus CPU time obtained with a bad initial guess ($\mathscr{I}_1$).

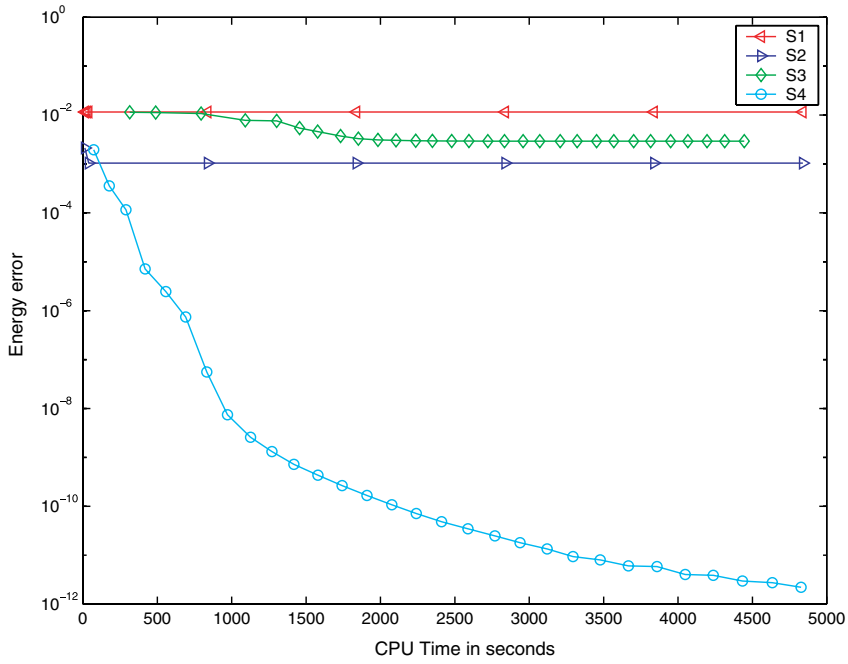


Fig. 8. Energy error versus CPU time obtained with a better initial guess ($\mathscr{I}_2$).

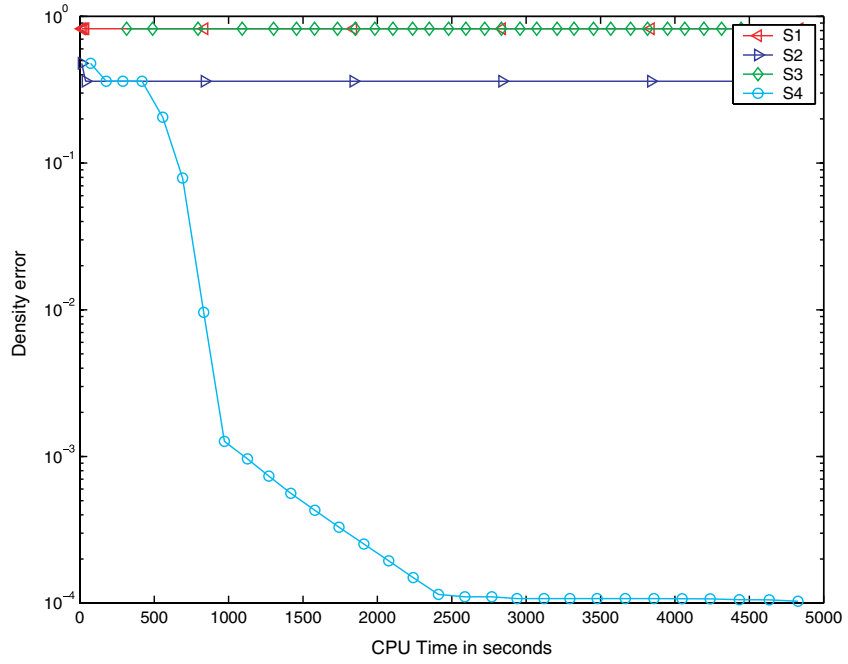


Fig. 9. Density error versus CPU time obtained with a better initial guess ($\mathscr{I}_2$).

The energy of the ground state of this matrix (i.e. the minimum of (1.6)) is $E_0 = -27663.484$. The number of blocks considered is $p = 100$. For the global step, we have collected these 100 blocks in 99 overlapping groups of 2 blocks. Interestingly, such a partition provides with optimal results regarding CPU time and memory requirement. It is observed in Figs. 6–9 that $\mathscr{S}_1$, $\mathscr{S}_2$ and $\mathscr{S}_3$ are not satisfactory for they converge towards

some local, non-global, minima of (2.2) whatever the initial guess. The failure of the strategy $\mathscr{S}_3$ performed on the initial guess $\mathscr{I}_2$ is surprising: this initial guess is not good enough. Indeed, if the initial guess is $\mathscr{I}_3$, we check numerically that the strategies $\mathscr{S}_3$ and $\mathscr{S}_4$ behave identically. Notice that the strategy $\mathscr{I}_3$ is identical to $\mathscr{S}_2$ applied to the initial guess $\mathscr{I}_2$.

We also remark that the strategy $\mathscr{S}_4$ performs very well whatever the initial guess (see Figs. 7 and 9). The same behavior is observed for the polymers $\mathscr{P}_2$ and $\mathscr{P}_3$. Finally, after orthonormalization, the density matrix minimization (DMM) method [24] failed with the random initial guess and reveals very slow with the initial guess $\mathscr{I}_2$. That is the reason why we consider the initial guess $\mathscr{I}_3$ to compare these methods.

4.3. Comparison with two other methods

Having emphasized the usefulness of all the ingredients of our MDD algorithm, we now compare it to two other algorithms:

- the diagonalization routine *dsbgv.f* from the LAPACK library;
- the density matrix minimization (DMM) method [24].

These two algorithms are seen as prototypical approaches for standard diagonalization algorithms and linear scaling techniques, respectively. They are only used here for comparison purposes. Regarding linear scaling methods, two other popular approaches are the Fermi Operator method [17] and the McWeeny iteration method [27]. We have observed that, at least in our own implementation, based on the literature, they are outperformed by the DMM method for the actual chemical systems we have considered. We therefore take DMM as a reference method for our comparison.

Recall that the routine *dsbgv.f* consists in the three-step procedure:

- transform the generalized eigenvalue problem into a standard eigenvalue problem by applying a Cholesky factorization to S;
- reduce the new matrix to be diagonalized to a tridiagonal form;
- compute its eigenelements by using the implicit QR method.

The algorithmic complexity of this approach is in N_b^3 and the required memory scales as N_b^2.

For the description of DMM method, we refer to [24]. Let us only mention here that this approach consists in a minimization procedure, applied to the energy expressed in terms of the density matrix. Both the algorithmic complexity and the memory needed for performing the DMM approach scale linearly with respect to the size N_b of the matrix. The DMM method is initialized with the density matrix $D = CC^t$ computed with the initial guess C of the domain decomposition method. Two important points for the tests shown below are the following.

First, we perform a cut-off on the coefficients on the various matrices manipulated throughout the calculation: only the terms of the density matrices within the frame defined in Fig. 4 are taken into account. Such a cut-off has some impact on the qualities of the results obtained with the DMM method. We are however not able to design a better comparison.

Second, the DMM method requires the knowledge of the Fermi level (as is the case for the linear scaling methods commonly used in practice to date). The determination of the Fermi level is the purpose of an outer optimization loop. In contrast, the MDD approach computes an approximation of the Fermi level at each iteration. Here, for the purpose of comparison, we *provide* DMM with the exact value of the Fermi level. Consequently, the CPU times for the DMM method displayed in the sequel are underestimated.

We emphasize that the routine *dsbgv.f* computes the entire spectrum of the matrix, both eigenvalues and eigenvectors. In contrast, the MDD approach only provides with the lowest N eigenvalues, among N_b, and the projector on the vector space spanned by the corresponding eigenvectors, not the eigenvectors themselves.

4.3.1. Comparison with direct diagonalization and DMM

We have computed the ground states of the polymers $\mathscr{P}_1$, $\mathscr{P}_2$ and $\mathscr{P}_3$ with the three methods (direct diagonalization, DMM and MDD) and for various numbers n_m of monomers, corresponding to matrix sizes N_b in the range 10^3–10^5.

For DMM and MDD, the initial guess is generated following the strategy $\mathscr{I}_3$.

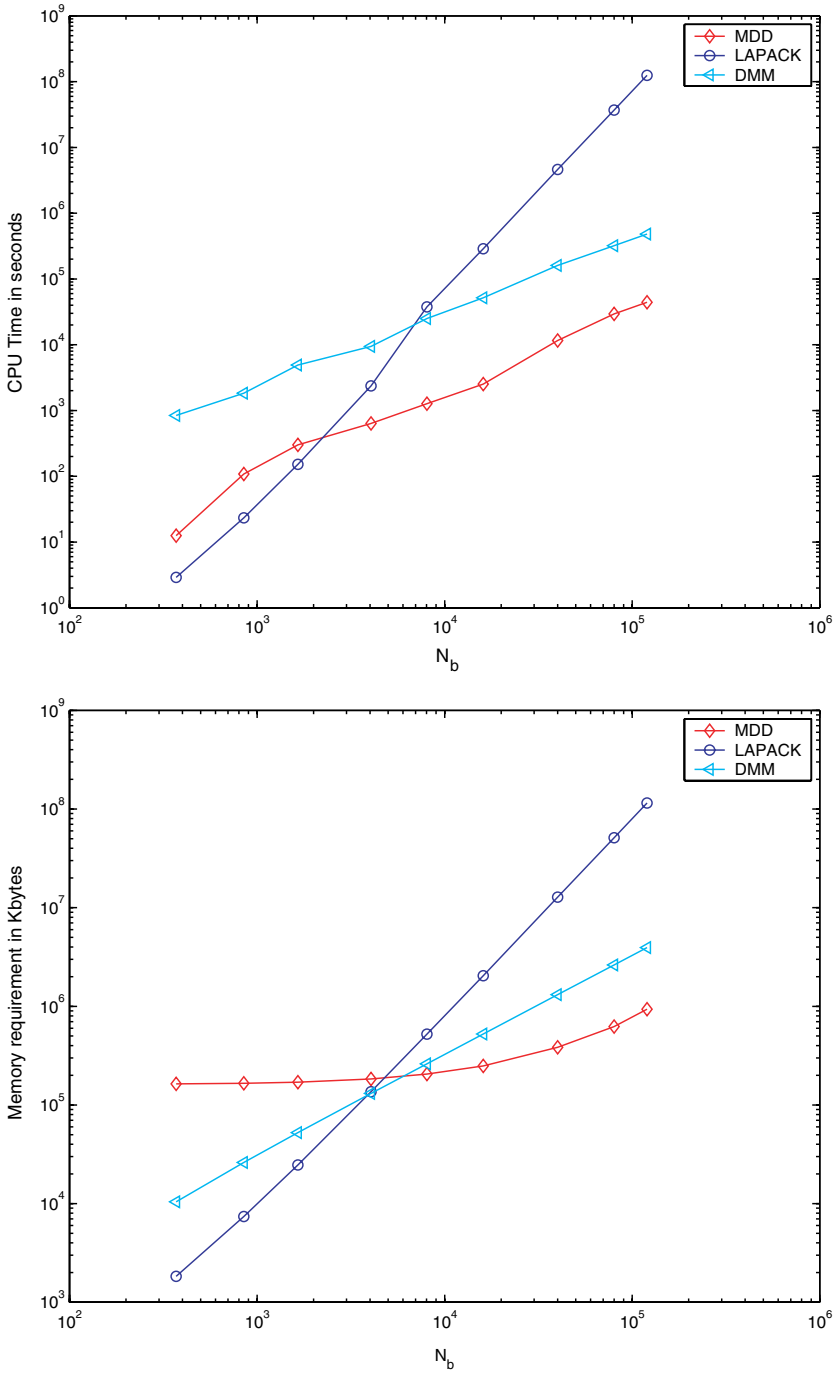


Fig. 10. Scaling of the CPU time (top) and memory requirement (bottom) for the polymer $\mathscr{P}_1$.

The results regarding the CPU time at convergence and the memory requirement are displayed in Figs. 10–12 for the polymers $\mathscr{P}_1$, $\mathscr{P}_2$, and $\mathscr{P}_3$, respectively.

For small values of N_b, i.e. up to around 10^4, the results observed for the direct diagonalization, DMM and MDD agree. The CPU times for our MDD approach scale linearly with N_b.

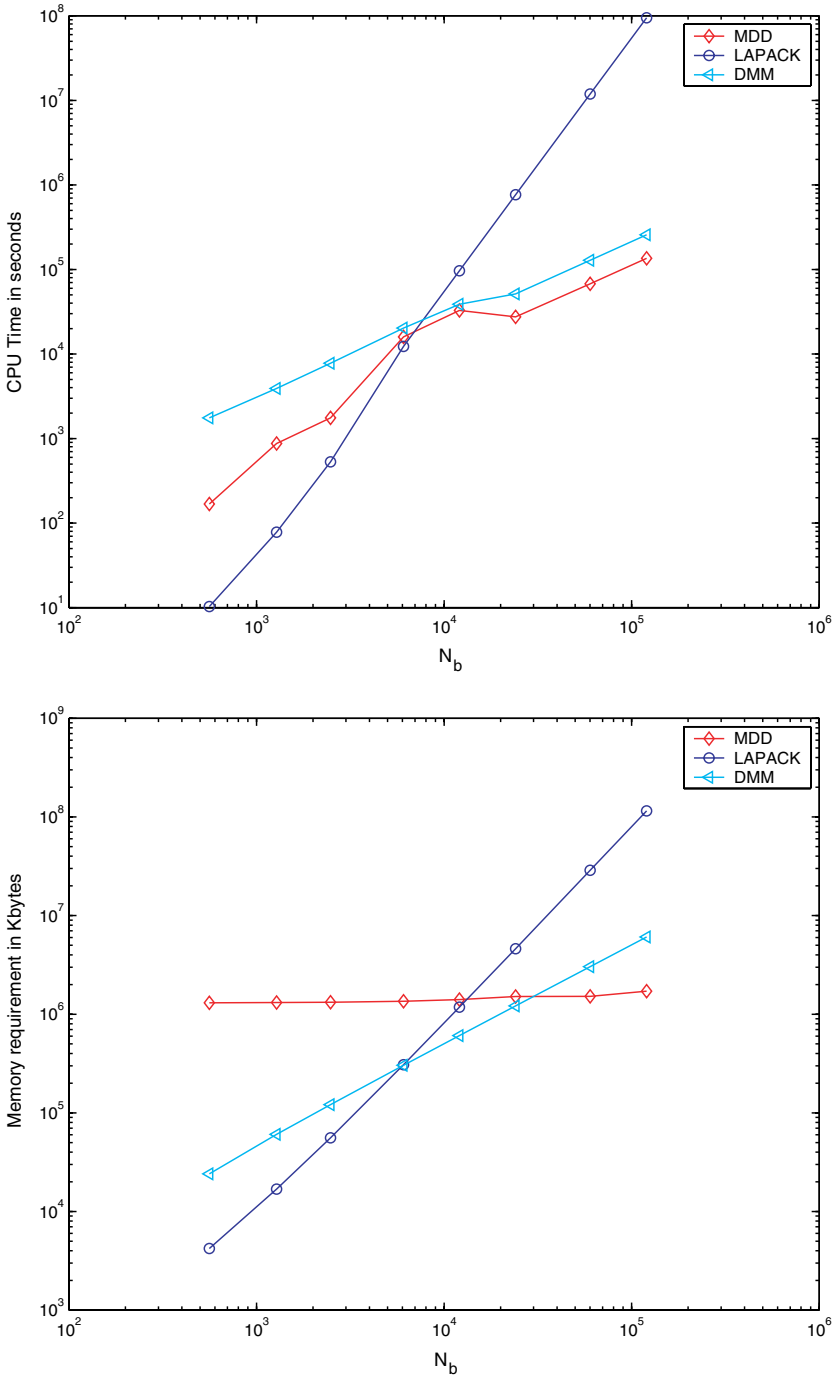


Fig. 11. Scaling of the CPU time (top) and memory requirement (bottom) for the polymer $\mathscr{P}_2$.

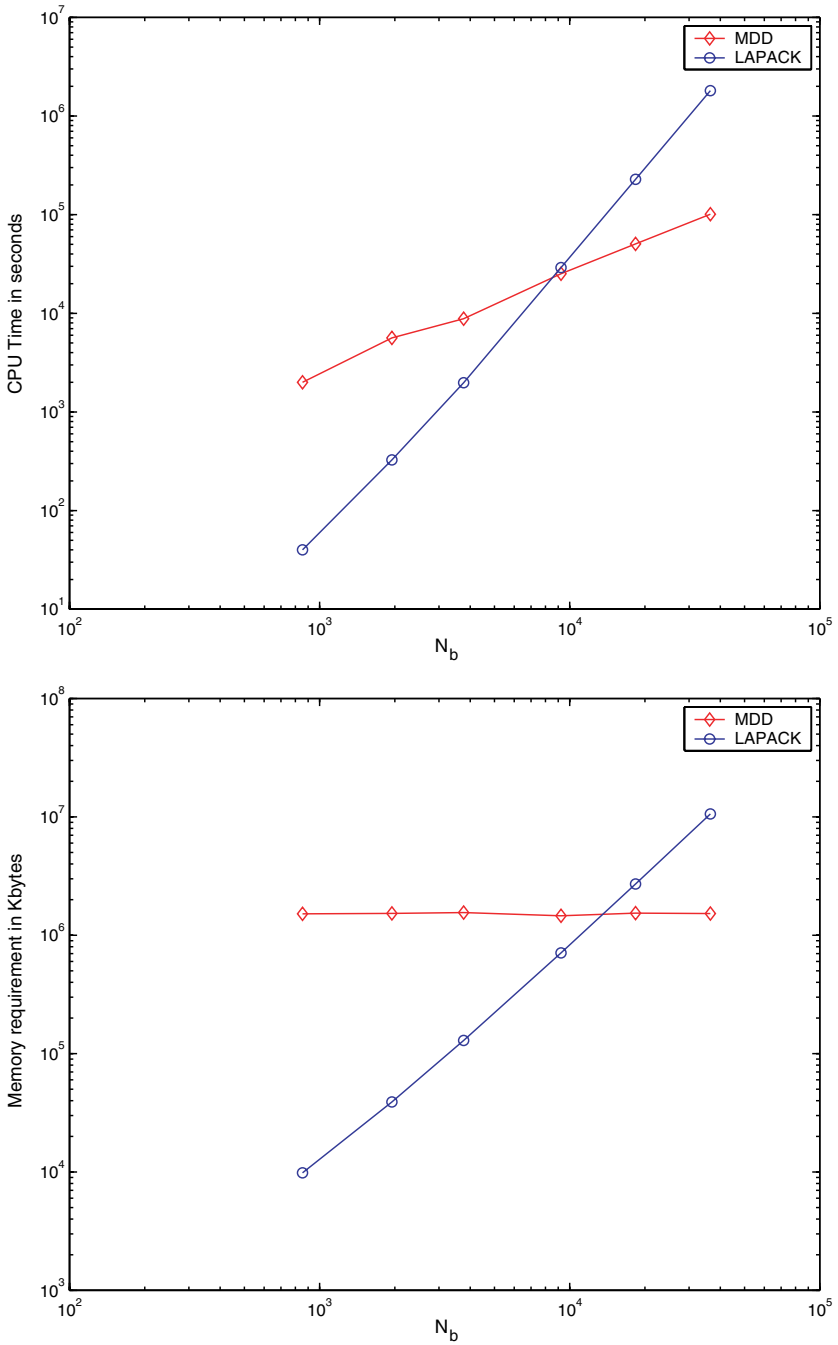


Fig. 12. Scaling of the CPU time (top) and memory requirement (bottom) for the polymer $\mathscr{P}_3$.

For larger values of N_b, the limited memory prevented us from either performing an exact diagonalization or from implementing DMM. So, we extrapolate the CPU time and memory requirement according to the scaling observed for smaller N_b.

The data for the DMM method are not plotted in Fig. 12 as the DMM method does not converge for the polymer $\mathscr{P}_3$ when the number of monomers exceeds 10^3. From our point of view, it comes from the truncation errors which cause the divergence of the method (note that the truncation strategy we consider here is very simple).

4.3.2. Comparison with DMM and a hybrid strategy

We now concentrate on the two approaches that scale linearly, namely DMM and MDD. We consider:

- $\mathscr{P}_1$ with 4001 monomers, corresponding to $N_b = 40050$,

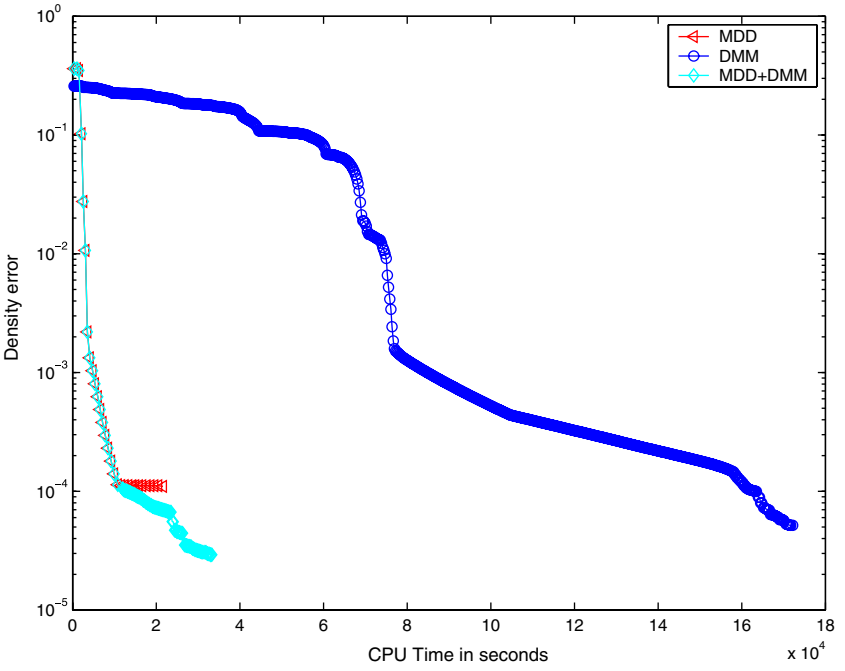


Fig. 13. Evolution of the density error with the CPU time for the polymer $\mathscr{P}_1$ made of 4001 monomers.

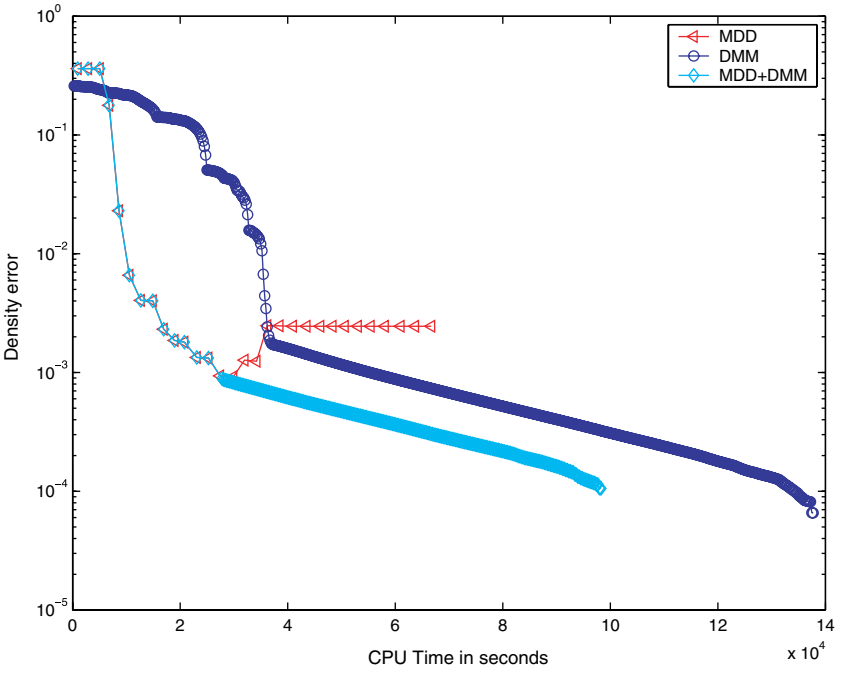


Fig. 14. Evolution of the density error with the CPU time for the polymer $\mathscr{P}_2$ made of 2404 monomers.

- $\mathscr{P}_2$ with 2404 monomers, corresponding to $N_b = 24080$,
- $\mathscr{P}_3$ with 208 monomers, corresponding to $N_b = 854$.

These particular values have been chosen for the purpose of having simple values for the numbers of blocks. For each of the three polymers, we compare the DMM and MDD methods initialized by the strategy $\mathscr{I}_3$ and a

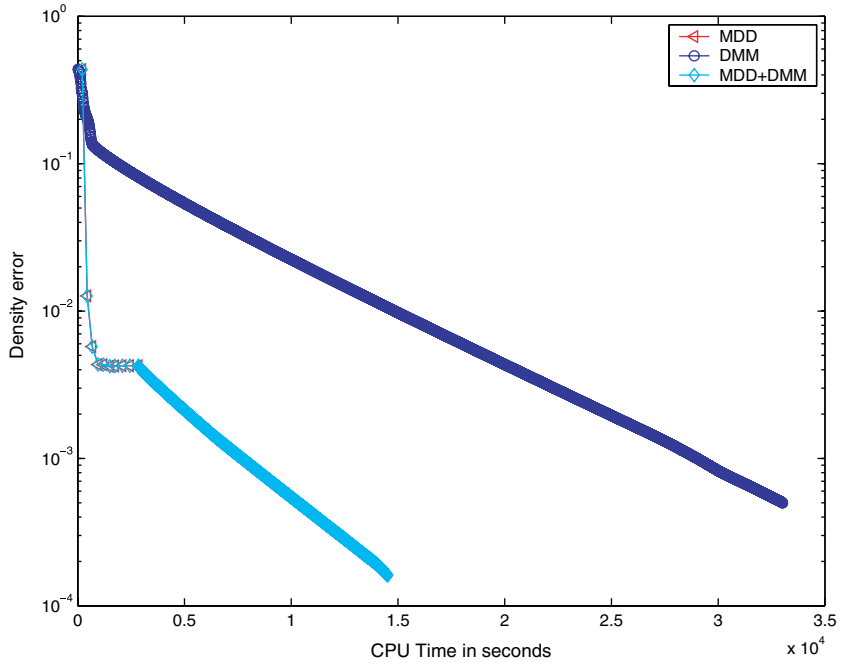


Fig. 15. Evolution of the density error with the CPU time for the polymer $\mathscr{P}_3$ made of 208 monomers.

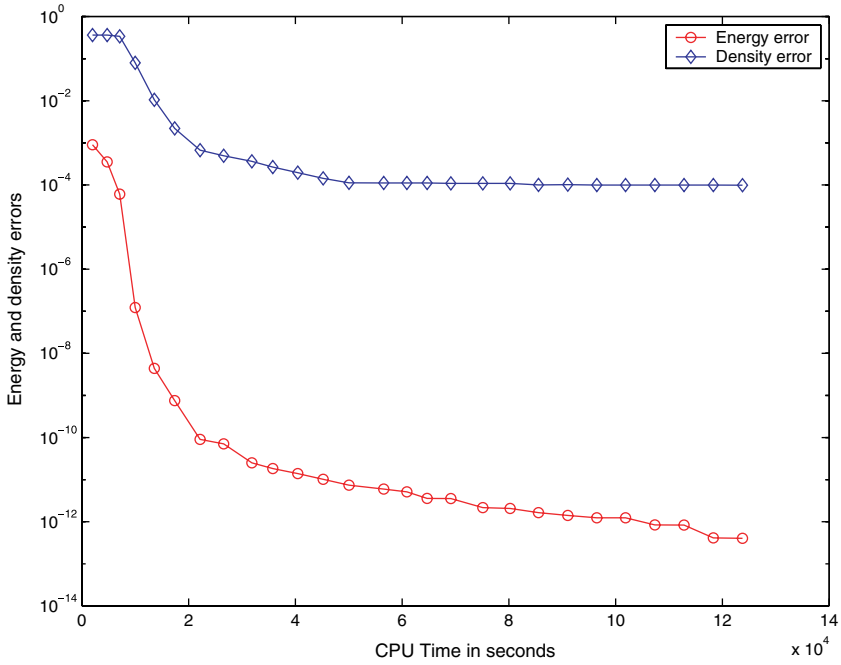


Fig. 16. Evolution of the MDD energy and density errors versus CPU time for the polymer $\mathscr{P}_1$ (20001 monomers, $N_b = 200050$).

hybrid strategy. The hybrid strategy consists of a certain number of iterations performed with MDD, until convergence is reached for this method, followed by iterations with DMM. We use the following stopping criterion for MDD:

$$\|D_n - D_{n-1}\| \geqslant \|D_{n-1} - D_{n-2}\| \quad \text{and} \quad \|D_n - D_{n-1}\| \leqslant \epsilon_a, \tag{3.19}$$

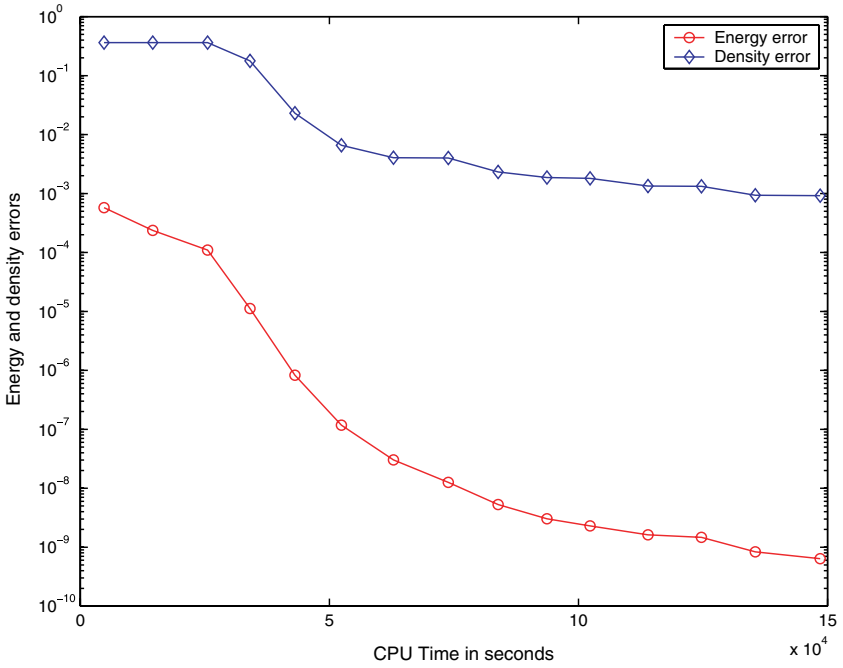


Fig. 17. Evolution of the MDD energy and density errors versus CPU time for the polymer $\mathscr{P}_2$ (12004 monomers, $N_b = 120080$).

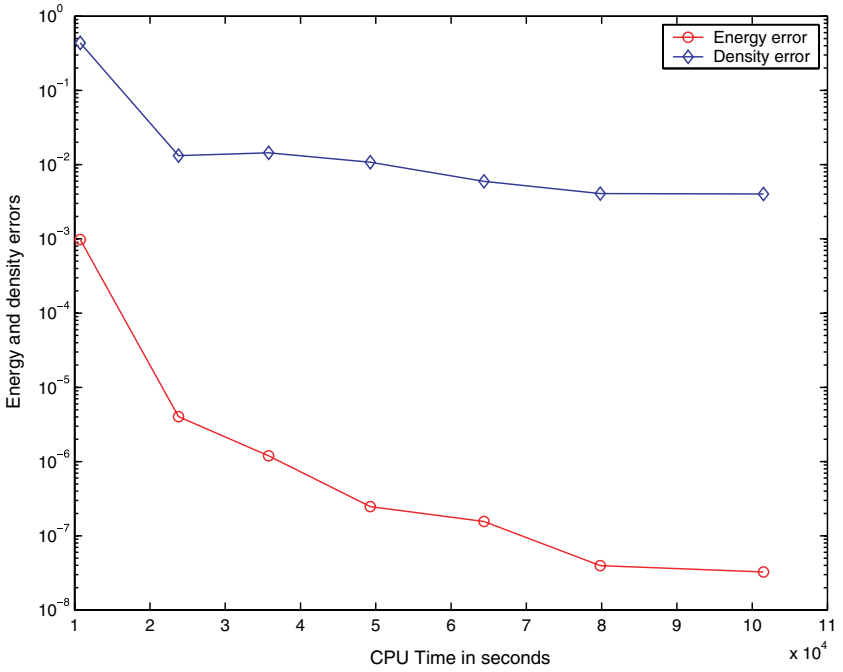


Fig. 18. Evolution of the MDD energy and density errors versus CPU time for the polymer $\mathscr{P}_3$ (5214 monomers, $N_b = 36526$).

where ϵ_a is a threshold parameter. We take $\epsilon_a = 10^{-4}$, respectively, $\epsilon_a = 10^{-3}$, for the polymer $\mathscr{P}_1$, respectively, $\mathscr{P}_2$ and $\mathscr{P}_3$.

Figs. 13–15 show the evolution of the error in density versus CPU time. The hybrid version is demonstrated to be a very efficient combination of the two algorithms.

For completeness, let us highlight the temporary increase for the error in density appearing in Fig. 14 when MDD is used on $\mathscr{P}_2$. Analogously, the energy of the current solution, which is actually below the reference energy, also increases. In fact, this is due to a loss of precision in the orthonormality constraints. In MDD, these constraints are not imposed exactly at each iteration, but only approximately (see Eq. (3.12)).

Finally, we report in Figs. 16–18 the results obtained with MDD for the largest possible case that can be performed on our platform, owing to memory limitation. We used the initial guesses obtained with the strategy $\mathscr{I}_3$. Notice that for the local step the memory requirement scales linearly with respect to the number n_m of monomers, while for the global step, the memory requirement is independent of n_m. Therefore, for large polymers, the memory needed by MDD is controlled by the local step. In contrast, for small polymers, the most demanding step in terms of memory is the global step.

5. Conclusions and remarks

The domain decomposition algorithm introduced above performs well, in comparison to the two standard methods considered. More importantly, our approach is an effective *preconditioning technique* for DMM iterations. Indeed, MDD provides a rapid and accurate approximation, both in terms of energy and density matrix, regardless of the quality of the initial guess. In contrast, DMM outperforms MDD when the initial guess is good, but only performs poorly, or may even diverge, when this is not the case. The combination of the two methods seems to be optimal. More generally, our MDD algorithm could constitute a good preconditioner to all variational methods, such as the orbital minimization method [26].

Regarding the comparison with DMM, the following comments are in order.

- All our calculations have been performed on a single processor machine. Potentially, both DMM and MDD should exhibit the same speed-up when parallelized. We therefore consider the comparison valid, at least qualitatively, for parallel implementations. The parallelization of the MDD is currently in progress, and hopefully will confirm the efficiency of the approach.
- We recall the Fermi level has to be provided to the DMM method. This is an additional argument in favor of the MDD approach.
- The MDD method, in contrast to the other linear scaling methods, does not perform any truncation in the computations. So, once the profile of C is choosen, the method does not suffer of any instabilities, contrary to DMM (or OM) for which divergences have been observed for the polymer $\mathscr{P}_3$.
- The domain decomposition method makes use of several threshold parameters. For the three polymers we have considered, the optimal values of these parameters, except for the stopping criterion ϵ_a (Eq. (3.19)), are the same. We do not know yet if this interesting feature is a general rule.
- Recall our method solves problem (2.2), which is only an approximation of problem (2.1). Therefore, the relative error obtained in the limit is only a measure of the difference between (2.2) and (2.1). In principle, such a difference could be made arbitrarily small by an appropriate choice of the parameters of problem (2.2).
- Finally, let us emphasize that there is much room for improvement in both the local and the global steps. We have designed an overall multilevel strategy that performs well, but each subroutine may be significantly improved. Another interesting issue is the interplay between the nonlinear loop in the Hartree–Fock or Kohn–Sham problems (self-consistent field – SCF – convergence [7,14,20]) and the linear subproblem considered in the present article. Future efforts will go in these directions.

Acknowledgments

We are indebted to Guy Bencteux (EDF) for valuable discussions and for his help in the implementation. C.L.B. and E.C. acknowledge many stimulating discussions with Richard Lehoucq (Sandia National Laboratories). The financial support of EDF (Electricité de France) is acknowledged.

References

[1] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, D. Sorensen, LAPACK Users' Guide, third ed., SIAM, 1999.

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

[4] N.W. Ashcroft, N.D. Mermin, Solid-State Physics, Saunders College Publishing, 1976.

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

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

[7] E. Cancès, M. Defranceschi, W. Kutzelnigg, C. Le Bris, Y. Maday, Computational quantum chemistry: a primerHandbook of Numerical Analysis, Special Volume, Computational Chemistry, vol. X, North-Holland, Amsterdam, 2003.

[8] S. Goedecker, L. Colombo, Efficient linear scaling algorithm for tight-binding molecular dynamics, Phys. Rev. Lett. 73 (1994) 122.

[9] S. Goedecker, M. Teter, Tight-binding electronic-structure calculations and tight-binding molecular dynamics with localized orbitals, Phys. Rev. B 51 (1995) 9455.

[10] F.R. Krajewski, M. Parrinello, Stochastic linear scaling for metals and nonmetals, Phys. Rev. B 71 (2005) 233105.

[11] F.R. Krajewski, M. Parrinello, Linear scaling electronic structure calculations and accurate statistical mechanics sampling with noisy forces, Phys. Rev. B 73 (2006) 041105(R).

[12] T. Ozaki, T. Terakura, Convergent recursive O(N) calculations for *ab initio* tight-binding, Phys. Rev. B 64 (2001) 195126.

[13] T. Ozaki, Linear scaling Krylov subspace method for large scale *ab initio* electronic structure calculations of metals, 2005, cond-mat/0509291.

[14] E. Cancès, C. Le Bris, Can we outperform the DIIS approach for electronic structure calculations, Int. J. Quantum Chem. 79 (2000) 82–90.

[16] P.M.W. Gill, Molecular integrals over Gaussian basis functions, Adv. Quantum Chem. 25 (1994) 141–205.

[17] S. Goedecker, Linear scaling electronic structure methods, Rev. Mod. Phys. 71 (1999) 1085–1123.

[18] W.J. Hehre, L. Radom, P.v.R. Schleyer, J.A. Pople, *Ab initio* Molecular Orbital Theory, Wiley, New York, 1986.

[19] U.L. Hetmaniuk, 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.

[20] K.N. Kudin, G.E. Scuseria, E. Cancès, A black-box self-consistent field convergence algorithm: one step closer, J. Chem. Phys. 116 (2002) 8255–8261.

[21] W. Kohn, Analytic properties of Bloch waves and Wannier functions, Phys. Rev. 115 (1959) 809–821.

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

[23] C. Le Bris, Computational chemistry from the perspective of numerical analysis, Acta Numer. 14 (2005) 363–444.

[24] X.-P. Li, R.W. Numes, D. Vanderbilt, Density-matrix electronic structure method with linear system size scaling, Phys. Rev. B 47 (1993) 10891–10894.

[25] R. McWeeny, Methods of Molecular Quantum Mechanics, second ed., Academic Press, New York, 1992.

[26] P. Ordejón, D.A. Drabold, M.D. Grumbach, R.M. Martin, Unconstrained minimization approach for electronic computations that scales linearly with system size, Phys. Rev. B 48 (1993) 14646–14649.

[27] A. Palser, D. Manopoulos, Canonical purification of the density matrix in electronic structure theory, Phys. Rev. B 58 (1998) 12704–12711.

[29] D. Sánchez-Portal, P. Ordejón, E. Artacho, J.M. Soler, Density-functional method for very large systems with LCAO basis sets, Int. J. Quantum Chem. 65 (1997) 453–461.

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

[31] C. Paige, M. Saunders, Solution of sparse indefinite systems of linear equations, SIAM J. Numer. Anal. 12 (1975) 617–629.

[32] 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, J.A. Pople, Gaussian 98 (Revision A.7), Gaussian Inc., Pittsburgh, PA, 1998.