Commun. Comput. Phys.
doi: 10.4208/cicp.101114.170615a

Vol. **19**, No. 3, pp. 632-647
March 2016

Numerical Methods for Solving the Hartree-Fock Equations of Diatomic Molecules II

John C. Morrison[1], Kyle Steffen[1], Blake Pantoja[1], Asha Nagaiya[1], Jacek Kobus[2,*] and Thomas Ericsson[3]

[1] *Department of Physics and Astronomy, University of Louisville, Louisville, KY, 40292, USA.*
[2] *Institute of Physics, Faculty of Physics, Astronomy and Informatics, Nicolaus Copernicus University, Grudziadzka 5, 87-100 Torun, Poland.*
[3] *Department of Mathematical Sciences, Chalmers University of Technology SE-412 96 Göteborg, Sweden*

Communicated by Michel A. Van Hove

Received 10 November 2014; Accepted (in revised version) 17 June 2015

Abstract. In order to solve the partial differential equations that arise in the Hartree-Fock theory for diatomic molecules and in molecular theories that include electron correlation, one needs efficient methods for solving partial differential equations. In this article, we present numerical results for a two-variable model problem of the kind that arises when one solves the Hartree-Fock equations for a diatomic molecule. We compare results obtained using the spline collocation and domain decomposition methods with third-order Hermite splines to results obtained using the more-established finite difference approximation and the successive over-relaxation method. The theory of domain decomposition presented earlier is extended to treat regions that are divided into an arbitrary number of subregions by families of lines parallel to the two coordinate axes. While the domain decomposition method and the finite difference approach both yield results at the micro-Hartree level, the finite difference approach with a 9-point difference formula produces the same level of accuracy with fewer points. The domain decomposition method has the strength that it can be applied to problems with a large number of grid points. The time required to solve a partial differential equation for a fine grid with a large number of points goes down as the number of partitions increases. The reason for this is that the length of time necessary for solving a set of linear equations in each subregion is very much dependent upon the number of equations. Even though a finer partition of the region has more subregions, the time for solving the set of linear equations in each subregion is very much smaller. This feature of the theory may well prove to be a decisive factor for solving the two-electron pair equation, which – for a diatomic molecule – involves solving partial differential equations with five independent variables. The domain decomposition theory also makes it possible to study complex molecules by dividing them into smaller fragments that

*Corresponding author. *Email address:* `jacek.kobus@fizyka.umk.pl` (J. Kobus)

 ©2016 Global-Science Press

https://doi.org/10.4208/cicp.101114.170615a Published online by Cambridge University Press

are calculated independently. Since the domain decomposition approach makes it possible to decompose the variable space into separate regions in which the equations are solved independently, this approach is well-suited to parallel computing.

AMS subject classifications: 65N35, 65N22, 65F10

Key words: Pair equation, Hartree-Fock equation, domain decomposition, Hermite splines, high-order finite differences, successive over-relaxation.

1 Introduction

The multiconfiguration Hartree-Fock equations (MCHF) provide an approximate description of atoms and molecules and serve as the starting point of more accurate calculations. Accurate numerical solutions of these equations for atoms have been reported by Fisher [1, 2]. Numerical solutions of the Hartree-Fock equations for diatomic molecules have been reported by McCullough [3], Heinemann and coworkers [4,5] and Laaksonen, Pykkö, and Sundholm [6], and more recently by Kobus, Laaksonen, and Sundholm [7,8].

The first article of this series [9] shows how the spline collocation and domain decomposition methods can be used to solve Poisson-like equations in two variables. This was a necessary step for solving the Hartree-Fock equations for diatomic molecules since the HF equations themselves are of the same form as the Poisson equation and the Coulomb and exchange potentials in the HF equations can be obtained by solving appropriate Poisson equations. The theory of domain decomposition presented earlier is extended to treat regions that are divided into an arbitrary number of subregions by vertical and horizontal interfaces. We test the approach by solving the equation for the He^+ system and compare the results with the numerical ones using finite differences.

In Section 2 of this article, we shall give the Hartree-Fock equations for a diatomic molecule in spheroidal coordinates. Section 3 introduces the domain decomposition algorithm and applies the algorithm to a model problem in which a region is divided into an arbitrary number of subregions by families of lines parallel to the two coordinate axes. Section 4 shows how the eigenvalues and eigenfunctions of the Hartree-Fock theory can be obtained using the inverse iteration and the inverse Arnoldi method or, within the frame-work of finite differences, by using the Raleigh quotient. In this section we shall also compare results obtained using spline collocation and domain decomposition with results obtained using finite differences and the successive over-relaxation (SOR) method.

2 The Hartree-Fock equations for a diatomic molecule

The Hartree-Fock equations for a diatomic molecule in atomic units can be written (cf. [9])

$$\left[-\frac{1}{2}\nabla^2-\frac{Z_A}{r_A}-\frac{Z_B}{r_B}+\sum_b(2J_b-K_b)\right]\psi_a(\mathbf{r})=\epsilon\psi_a(\mathbf{r}), \tag{2.1}$$

https://doi.org/10.4208/cicp.101114.170615a Published online by Cambridge University Press

where r_A and r_B denote the distances of the electron from the two nuclei separated by R atomic units, and the charges of the two nuclei are denoted by Z_A and Z_B. The Coulomb interaction among the electrons is described by the direct and exchange potentials J_b and K_b which are solutions of the equations

$$\begin{aligned}\nabla^2 J_b &= -4\pi\psi_b^*\psi_b,\\ \nabla^2 K_b\psi_a &= -4\pi\psi_b^*\psi_a.\end{aligned} \tag{2.2}$$

Diatomic molecules are usually described in spheroidal coordinates

$$\begin{aligned}&\xi=(r_a+r_b)/R, && \text{where } 1\le\xi<\infty,\\ &\eta=(r_a-r_b)/R, && \text{where } -1\le\eta\le 1,\\ &\phi, && \text{where } 0\le\phi\le 2\pi.\end{aligned} \tag{2.3}$$

One can ensure that the grid points are clustered near the nuclei where the wave functions change most rapidly by making a variable transformation

$$\begin{aligned}\eta&=\cos\nu \quad && \text{for } 0\le\nu\le\pi,\\ \xi&=\cosh\mu \quad && \text{for } 0\le\mu\le\infty.\end{aligned} \tag{2.4}$$

The Hartree-Fock equations may then be written as

$$\begin{aligned}&-\sin\nu\sinh\mu\frac{\partial^2 u_a}{\partial\nu^2}-\cos\nu\sinh\mu\frac{\partial u_a}{\partial\nu}-\sin\nu\sinh\mu\frac{\partial^2 u_a}{\partial\mu^2}-\sin\nu\cosh\mu\frac{\partial u_a}{\partial\mu}\\ &\quad+\sin\nu\sinh\mu\left(\frac{m^2}{\xi^2-1}+\frac{m^2}{1-\eta^2}\right)u_a\\ &\quad+\sin\nu\sinh\mu\left[-RZ\xi+R\Delta\eta+\frac{R^2}{2}(\xi^2-\eta^2)\sum_b(2J_b-K_b)\right]u_a\\ &=\sin\nu\sinh\mu\frac{R^2}{2}(\xi^2-\eta^2)\epsilon u_a,\end{aligned} \tag{2.5}$$

where $Z=Z_A+Z_B$ and $\Delta=Z_A-Z_B$ and $\psi_a=e^{im_a\phi}u_a$. Eq. (2.5) is obtained by using the coordinates (2.3) and the substitution (2.4) and multiplying the resulting equation by $\sin\nu\sinh\mu$ to make the equation self-adjoint. A differential operator is self-adjoint if the derivatives of the coefficients of the second derivatives are equal to the coefficients of the first derivatives and if the function satisfies homogeneous Dirichlet or von Neumann boundary conditions. The above differential operator has these properties.

3 The domain decomposition algorithm

As in the previous article of this series [9], we shall describe calculations using a basis of Hermite splines to represent wave functions and to convert partial differential equations

https://doi.org/10.4208/cicp.101114.170615a Published online by Cambridge University Press

(PDEs) into equations involving matrices. The numerical results given in the previous paper and the results we give here show that these methods can give a high level of numerical accuracy.

The main difficulty one encounters in applying numerical methods to realistic problems in quantum chemistry is that the time required rises precipitously as the number of grid points increases. The domain decomposition method described in this Section enables one to reduce the time necessary to solve PDEs by breaking the entire domain into smaller regions where the equations are solved independently. The domain decomposition method also makes it possible to take on much larger problems such as the pair equation that would be intractable with ordinary methods. In our previous paper [9], we divided the entire region into nine subregions and solved a PDE in the large central region using the efficient preconditioned conjugate gradient method and solved the PDE in the narrow regions near the boundary with a banded Gaussian solver. In this paper, we apply domain decomposition more aggressively by breaking the entire region into an arbitrary number of subregions with lines parallel to the two coordinate axes.

Because the domain decomposition method is central to the results we shall present in this paper, we would now like to describe the algorithm for a simple problem of a region divided into two subregions by a vertical interface. A derivation of the algorithm for this simple case can be found in the first article of this series [9] and in earlier articles on the spline collocation method [10] and [11].

A region that is divided into two regions by a vertical interface is illustrated in Fig. 1. We denote a PDE by

$$\mathbf{L}u(x,y)=f(x,y) \quad \text{for } a\le x\le b,\ c\le y\le d, \tag{3.1}$$

where $\mathbf{L}$ is a differential operator and $u(x,y)$ and $f(x,y)$ are functions of two variables. As shown in Fig. 1, we shall denote the two regions by Ω_1 and Ω_2 and the interface between the two regions by Γ. The solutions in these two regions will be denoted by $u_1(x,y)$ and $u_2(x,y)$, and the values of the grid points on the interface between the two regions will be denoted by γ.

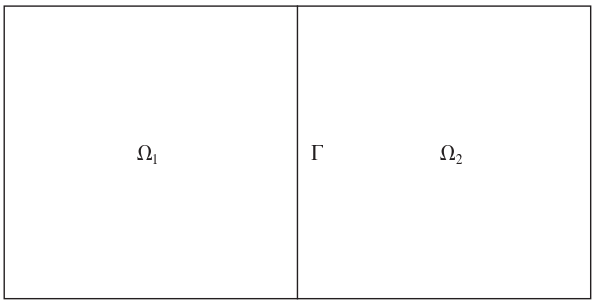


Figure 1: Region divided into 2 subregions by interface.

The domain decomposition method can be applied to the problem of solving PDEs with either Dirichlet or Neumann boundary conditions. Here, we shall suppose that the

https://doi.org/10.4208/cicp.101114.170615a Published online by Cambridge University Press

solution of Eq. (3.1) satisfies homogeneous Neumann conditions on the left and right boundaries and on the lower boundary while satisfying a homogeneous Dirichlet condition along the upper boundary

$$\begin{aligned} &\mathcal{D}_x u(a,y)=0, \quad y\in(c,d), \qquad \mathcal{D}_x u(b,y)=0, \quad y\in(c,d), \\ &\mathcal{D}_y u(x,c)=0, \quad x\in[a,b], \qquad u(x,d)=0, \qquad x\in[a,b], \end{aligned} \tag{3.2}$$

where $\mathcal{D}_x$ and $\mathcal{D}_y$ represent partial derivatives with respect to the x- and y-coordinates.

To use the domain decomposition method to solve Eq. (3.1) in the entire region with $a\le x\le b$ and $c\le y\le d$, we must find the value of the solution along the interface. Once u is known on the interface, the functions u_1 and u_2 in the two regions can be found independently. The solution of a partial differential equations in two dimensions is uniquely defined once the boundary conditions are specified. This can be proved mathematically by showing that the number of boundary conditions and collocation conditions is equal to the number of spline coefficients because then the discrete form of the differential equation can be solved exactly.

An equation for the solution on the interface can be obtained by resolving the function u into two functions $\hat{u}$ and $\tilde{u}$, which satisfy the same conditions as u along the exterior boundary

$$u(x,y)=\hat{u}(x,y)+\tilde{u}(x,y). \tag{3.3}$$

The properties of $\hat{u}$ and $\tilde{u}$ are complementary with $\hat{u}$ satisfying the same PDE as u but being equal to zero on the interface between the two regions and $\tilde{u}$ satisfying the homogeneous form of the PDE satisfied by u and is equal to u on the interface between the two regions. The function $\hat{u}_1$ satisfies the equation

$$\mathbf{L}\hat{u}_1(x,y)=f(x,y) \quad \text{for } a\le x\le\gamma,\ c\le y\le d, \tag{3.4}$$

with boundary conditions

$$\begin{aligned} &\mathcal{D}_x\hat{u}_1(a,y)=0, \quad y\in(c,d), \qquad \mathcal{D}_y\hat{u}_1(x,c)=0, \quad x\in[a,\gamma], \\ &\hat{u}_1(x,d)=0, \qquad x\in[a,\gamma], \qquad \hat{u}_1(\gamma,y)=0, \qquad y\in(c,d), \end{aligned} \tag{3.5}$$

while the function $\hat{u}_2$ satisfies the equation

$$\mathbf{L}\hat{u}_2(x,y)=f(x,y) \quad \text{for } \gamma\le x\le b,\ c\le y\le d, \tag{3.6}$$

with boundary conditions

$$\begin{aligned} &\mathcal{D}_x\hat{u}_2(b,y)=0, \quad y\in(c,d), \qquad \mathcal{D}_y\hat{u}_2(x,c)=0, \quad x\in[\gamma,b], \\ &\hat{u}_2(x,d)=0, \qquad x\in[\gamma,b], \qquad \hat{u}_2(\gamma,y)=0, \qquad y\in(c,d). \end{aligned} \tag{3.7}$$

Since the functions $\hat{u}_1$ and $\hat{u}_2$ satisfy boundary conditions that do not depend upon the nature of the solution along the interface, these functions can immediately be determined

https://doi.org/10.4208/cicp.101114.170615a Published online by Cambridge University Press

by solving the PDEs in the two regions. So, the functions $\hat{u}_1$ and $\hat{u}_2$ may be regarded as known functions.

The function $\tilde{u}_1$ satisfies the equation

$$\mathbf{L}\tilde{u}_1(x,y)=0 \quad \text{for } a\le x\le\gamma,\ c\le y\le d, \tag{3.8}$$

and the function $\tilde{u}_2$ satisfies the equation

$$\mathbf{L}\tilde{u}_2(x,y)=0 \quad \text{for } \gamma\le x\le b,\ c\le y\le d. \tag{3.9}$$

The conditions satisfied by $\tilde{u}_1$ and $\tilde{u}_2$ along the exterior boundaries are identical to the conditions satisfied $\hat{u}_1$ and $\hat{u}_2$. On the interface between the two regions $\tilde{u}_1$ and $\tilde{u}_2$ are equal to the solution of Eq. (3.1).

To obtain the solution u along the interface, we define an interface operator K by the equation

$$Kv(y)=W_\Gamma, \tag{3.10}$$

where the spline function $v(y)$ defined for $c\le y\le d$ satisfies the boundary conditions, $v'(c)=0$ and $v(d)=0$. The function W_Γ is defined by the equation

$$W_\Gamma(\xi)=\mathcal{D}_xV_1(\gamma,\xi)-\mathcal{D}_xV_2(\gamma,\xi), \quad \text{for } \xi\in\mathcal{G}. \tag{3.11}$$

Here ξ is a member of the set of grid points on the interface. The function V_1 satisfies the equation

$$\mathbf{L}V_1(x,y)=0 \quad \text{for } a\le x\le\gamma,\ c\le y\le d \tag{3.12}$$

with boundary conditions

$$\begin{aligned} &\mathcal{D}_xV_1(a,y)=0, \quad y\in(c,d), \quad &&\mathcal{D}_yV_1(x,c)=0, \quad x\in[a,\gamma],\\ &V_1(x,d)=0, \quad x\in[a,\gamma], \quad &&V_1(\gamma,y)=v(y), \quad y\in(c,d), \end{aligned} \tag{3.13}$$

and the function V_2 satisfies the equation

$$\mathbf{L}V_2(x,y)=0 \quad \text{for } \gamma\le x\le b,\ c\le y\le d \tag{3.14}$$

with boundary conditions

$$\begin{aligned} &\mathcal{D}_xV_2(b,y)=0, \quad y\in(c,d), \quad &&\mathcal{D}_yV_2(x,c)=0, \quad x\in[\gamma,b],\\ &V_2(x,d)=0, \quad x\in[\gamma,b], \quad &&V_2(\gamma,y)=v(y), \quad y\in(c,d). \end{aligned} \tag{3.15}$$

For a particular spline function $v(y)$ defined for $c\le y\le d$, the boundary conditions for the functions $V_1(x,y)$ and $V_2(x,y)$ are well defined and the function $W_\Gamma(\xi)$ is given by Eq. (3.11). In the first article of this series we showed that if the function $v(y)$ is equal to the function u_Γ, which is the value of the solution of Eq. (3.1) along the interface, then the interface equation may be written

$$Ku_\Gamma=\mathcal{D}_x\hat{u}_2(\gamma,y)-\mathcal{D}_x\hat{u}_1(\gamma,y), \tag{3.16}$$

and we may define the following algorithm for solving the PDE (3.1) with boundary conditions (3.2):

https://doi.org/10.4208/cicp.101114.170615a Published online by Cambridge University Press

Step 1. Using $\hat{u}_1$ and $\hat{u}_2$ obtained by solving Eqs. (3.4)-(3.7), compute the right-hand side of Eq. (3.16).

Step 2. Solve Eq. (3.16) to find the solution of Eq. (3.1) along the interface.

Step 3. Solve Eqs. (3.8) and (3.9) to obtain the functions $\tilde{u}_1$ and $\tilde{u}_2$ and add these functions to $\hat{u}_1$ and $\hat{u}_2$ to obtain the solution u in regions Ω_1 and Ω_2.

The domain decomposition algorithm presented in the article describes how one can find the solution of a differential equation in each subregion by combining the $\hat{u}$ and $\tilde{u}$ solutions. The solution in all of space is thus obtained by solving the differential equation in each of the subregions into which the domain has been divided.

The domain decomposition methods can be tested by using the states of a single-electron atom or molecular ion. Equations for these states can be obtained by setting the direct and exchange terms equal to zero in Eq. (2.5). A simple model problem is provided by the singly ionized helium ion He^+ whose wave functions can be expressed analytically. For He^+, Z_A can be taken to be two and Z_B can be taken to be zero ($Z = Z_A + Z_B = 2$ and $\Delta = Z_A - Z_B = 2$). For diatomic molecules, the interatomic distance R is typically in the range between $2-6$ atomic units. The spheroidal coordinate system may thus be used to describe atoms and diatomic molecules in a unified way.

Using spheroidal coordinates defined by Eq. (2.3) and the coordinate transformation (2.4), the wave function for the ground state of a He^+ ion centered at A can be written

$$u(\nu,\mu) = \left(\frac{Z_A^3}{\pi}\right)^{1/2} \exp\left[-\frac{1}{2}RZ_A(\cos\nu + \cosh\mu)\right]. \tag{3.17}$$

This wave function satisfies homogeneous Neumann boundary conditions on the vertical sides and on the lower boundary while satisfying homogeneous Dirichlet boundary condition on a sufficiently high upper boundary.

We tested the accuracy of the domain decomposition theory for solving PDEs by evaluating the right-hand side of Eq. (2.5) at the grid points and comparing the solution of the equation with the analytical solution (3.17). For the spline collocation method used here and in our previous article, each coordinate axis is divided up into small subintervals that each contain two grid points called collocation points. Table 1 gives the maximum difference of the numerical solution and analytic function in the each of 48 subregions obtained by dividing the original region into 4 pieces for the first variable and 6 pieces for the second variable.

The accuracy of the solution of a PDE using third-order Hermite splines goes as h^4 and is independent of how the region is partitioned into subregions. The error of the solution of the PDE and error of the derivatives of the solution with respect to ν and μ are given in Table 2 for a number of different grids.

https://doi.org/10.4208/cicp.101114.170615a Published online by Cambridge University Press

Table 1: The maximum error of the function u and its partial derivatives u_ν and u_μ in a region divided into 4×6 subregions by vertical and horizontal interfaces. The grid used for the calculation consisted of 48 intervals in each direction.

k_μ	k_ν	u	u_ν	u_μ
1	1	2.6×10^{-7}	1.5×10^{-7}	9.6×10^{-8}
1	2	5.0×10^{-7}	8.2×10^{-7}	2.8×10^{-7}
1	3	2.1×10^{-6}	3.8×10^{-6}	5.9×10^{-7}
1	4	7.3×10^{-6}	4.6×10^{-6}	2.9×10^{-6}
2	1	1.4×10^{-7}	2.2×10^{-7}	2.9×10^{-7}
2	2	3.0×10^{-7}	7.3×10^{-7}	5.1×10^{-7}
2	3	8.4×10^{-7}	1.1×10^{-6}	1.1×10^{-6}
2	4	1.2×10^{-6}	7.1×10^{-7}	2.8×10^{-6}
3	1	1.5×10^{-7}	2.6×10^{-7}	3.7×10^{-7}
3	2	3.2×10^{-7}	3.3×10^{-7}	7.7×10^{-7}
3	3	5.6×10^{-7}	4.4×10^{-7}	1.5×10^{-6}
3	4	1.2×10^{-6}	7.0×10^{-7}	2.2×10^{-6}
4	1	1.6×10^{-7}	1.5×10^{-7}	5.3×10^{-7}
4	2	1.5×10^{-7}	1.8×10^{-7}	7.7×10^{-7}
4	3	4.0×10^{-7}	4.2×10^{-7}	1.6×10^{-6}
4	4	6.0×10^{-7}	4.2×10^{-7}	2.3×10^{-6}
5	1	1.0×10^{-7}	8.5×10^{-8}	5.3×10^{-7}
5	2	9.1×10^{-8}	9.5×10^{-8}	6.4×10^{-7}
5	3	2.3×10^{-7}	2.3×10^{-7}	1.6×10^{-6}
5	4	3.5×10^{-7}	2.3×10^{-7}	2.3×10^{-6}
6	1	7.1×10^{-8}	3.7×10^{-8}	4.7×10^{-7}
6	2	6.3×10^{-8}	6.9×10^{-8}	4.7×10^{-7}
6	3	1.6×10^{-7}	1.6×10^{-7}	1.2×10^{-6}
6	4	2.5×10^{-7}	1.6×10^{-7}	1.8×10^{-6}

Table 2: The maximum error of the function u and its partial derivatives with respect to ν and μ for grids with different numbers of intervals for each variable. The number of intervals in the ν- and μ-direction is denoted by n_1 and n_2, respectively.

$n_1\times n_2$	u	u_ν	u_μ	$u_{\nu\mu}$
48×48	7.3×10^{-6}	4.6×10^{-6}	2.9×10^{-6}	1.55×10^{-6}
96×96	5.7×10^{-7}	3.4×10^{-7}	2.2×10^{-7}	9.61×10^{-8}
192×192	4.3×10^{-8}	2.5×10^{-8}	1.6×10^{-8}	5.99×10^{-9}

https://doi.org/10.4208/cicp.101114.170615a Published online by Cambridge University Press

Table 3: CPU times (in seconds) for different numbers of intervals and different partitions of the entire domain. As in Table 2, n_1 and n_2 give the number of intervals in the ν- and μ-direction. Column two gives the number of subregions into which the entire domain is divided. Column three gives the time taken to form and factor the interface matrix, and the last column gives the time to solve the PDE once the interface matrix is factored.

$n_1 \times n_2$	Partition	Form & Factor Interface Matrix	Solve HF Equation
48×48	6×6	2.74	0.10
48×48	12×12	3.34	0.11
48×48	24×24	16.90	0.22
96×96	6×6	31.28	0.62
96×96	12×12	24.89	0.50
96×96	24×24	119.64	0.85
192×192	6×6	532.49	5.16
192×192	12×12	230.21	2.79
192×192	24×24	901.83	3.72

While the accuracy of the solution depends only upon the number of grid points, the time to solve the PDE depends upon the number of grid points and also the way the entire region is partitioned into subregions. We have formulated the theory so that the interface matrix is independent of the boundary conditions and the right-hand side of the PDE. The CPU times for calculations performed with different numbers of grid points and different partitioning of the entire region into subregions is shown in Table 3.

From the results given in Table 3, one can see that grids with many intervals and grid points in each direction can be more effectively solved by partitioning the entire region into a large number of subregions. While a grid with only 6 partitions in each of the two coordinate directions is competitive for a grid having only 48 intervals in each direction, a 6×6 partitioning of a problem with 192 intervals in each direction would clearly be less effective than a 12×12 or 24×24 partitioning. Our calculation for a region with 192 subintervals in the two coordinate directions with a partitioning of 12×12 subregions took 230.2 seconds to form and factor the interface equation and 2.79 seconds to solve the PDE. Since a 12×12 partitioning involves 144 subregions, the program only took $2.79/144 = 1.93\times10^{-2}$ seconds to solve the PDE in each of the subregions.

Parallel computing makes it possible to assign a few subregions to each core of a parallel machine, and hence makes calculations with a large number of subregions more attractive. Of course there is no need to write a parallel program to solve PDEs requiring a few seconds on a single CPU because of the extra overhead due to inter-region communication time involved in the calculation. By contrast, the two electron pair equation for a diatomic molecule will take much longer to solve and is expected to be much more heavily dependent upon parallel computing.

The domain decomposition algorithm can be used to solve Poisson-like equations that

https://doi.org/10.4208/cicp.101114.170615a Published online by Cambridge University Press

are found in the Hartree-Fock theory of diatomic molecules (cf. [8]). For such problems the interface matrices are not very large and can be constructed using Eq. (3.10), and the interface equation (3.16) can be solved using LAPACK routines. This is the approach we have used for the domain decomposition results presented in this article. For larger problems such as the two-electron pair equations considered in a forthcoming article, the interface equations can be solved iteratively using the generalized minimal residual method (GMRES) method without ever constructing the very large interface matrix.

The spline collocation method with third-order Hermite splines and the finite-difference method both provide the possibility of carrying out very accurate calculations of the properties of diatomic molecules. The spline representation errors go as h^4, which means that doubling the number of grid points causes the error to decrease by a factor of sixteen. Our earlier calculations show this [9]. Given enough grid points either approach can attain an arbitrary level of accuracy. However, because the finite-difference formulas for discretizing the differential equations and evaluating integrals are of higher-order, one would expect the finite-difference formalism to provide the same level of accuracy with fewer points. We should also note that Bialecki and Fairweather [10] have developed collocation methods using higher-order splines, which makes it possible to obtain a high level of accuracy with fewer points.

4 The eigenvalue problem

To implement the Hartree-Fock theory for diatomic molecules, one must solve PDE and eigenvalue equations. One typically makes an initial estimate of the wave functions and uses these wave functions to form the right-hand side of Poisson-like equations that can be solved to evaluate the terms in the Hartree-Fock equations corresponding to the interactions between the electrons. The Hartree-Fock equations themselves are eigenvalue equations. A modern description of iterative methods for solving eigenvalue problems can be found in the book by Heath [12] and Trefethen and Bau [13].

The Hartree-Fock equations (2.5) can be written

$$\mathbf{L}u = \epsilon D(\nu,\mu)u, \tag{4.1}$$

where $D(\nu,\mu) = \sin\nu\sinh\mu(R^2/2)(\xi^2-\eta^2)$. Because of the appearance of the function $D(\nu,\mu)$ on the right-hand side, Eq. (4.1) is regarded as a generalized eigenvalue equation.

The discrete form of Eq. (4.1) can be written

$$[L][u] = \epsilon[D][K][u], \tag{4.2}$$

where $[L]$ is the matrix corresponding to the operator $\mathbf{L}$ and $[u]$ is a vector whose components are the spline coefficients of the solution. $[D]$ is a diagonal matrix with the values of function $D[\nu,\mu]$ at the collocation points along the diagonal and the product of $[K]$ and $[u]$ gives the values of the function u at the collocation points. (See [9] for details.)

https://doi.org/10.4208/cicp.101114.170615a Published online by Cambridge University Press

To solve Eq. (4.2) using the inverse-iteration method or the inverse-Arnoldi method, one first subtracts a term $\sigma[D][K][u]$ from each side of the eigenvalue equation (4.2) to obtain

$$([L]-\sigma[D][K])[u]=(\epsilon-\sigma)[D][K][u], \tag{4.3}$$

where the constant σ is typically chosen to be slightly less that the eigenvalue of interest. The eigenvalue, $\epsilon-\sigma$ will then be the smallest eigenvalue and the inverse power method or the inverse-Arnoldi method can be expected to converge rapidly.

Eq. (4.3) can be simplified using the transformations

$$[v]=[K][u], \tag{4.4}$$

$$[w]=[D]^{1/2}[v], \tag{4.5}$$

to obtain the following form of the shifted eigenvalue equation

$$[D]^{-1/2}([L]-\sigma[D][K])[K]^{-1}[D]^{-1/2}[w]=(\epsilon-\sigma)[w]. \tag{4.6}$$

Since our intention is to use an inverse method, we invert this last equation to obtain

$$[D]^{1/2}[K]([L]-\sigma[D][K])^{-1}[D]^{1/2}[w]=\left(\frac{1}{\epsilon-\sigma}\right)[w]. \tag{4.7}$$

Eqs. (4.6) and (4.7) are in the form of simple eigenvalue equations. By introducing the operator

$$[A]=[D]^{1/2}[K]([L]-\sigma[D][K])^{-1}[D]^{1/2}, \tag{4.8}$$

the inverted eigenvalue equation (4.7) may be written simply as

$$[A][w]=\left(\frac{1}{\epsilon-\sigma}\right)[w]. \tag{4.9}$$

The Arnoldi method, which we shall use to solve Eq. (4.7), relies upon constructing an orthogonal basis in the Krylov space with the following modified Gram-Schmidt algorithm:

---

b = arbitrary, $q_1=b/||b||$

for $n=1,2,3,\cdots$

$v=[A]q_n$

for $j=1$ **to** n

$h_{jn}=q_j^* v$

$v=v-h_{jn}q_j$

https://doi.org/10.4208/cicp.101114.170615a Published online by Cambridge University Press

$$h_{n+1,n} = ||v||$$

$$q_{n+1} = ||v||/h_{n+1,n}$$

The algorithm generates an orthogonal basis $\{\mathbf{q}_n\}$. These basis vectors form the columns of an orthogonal matrix $[Q_n]$ that reduces the inverse matrix $[A]$ to Hessenberg form. The matrix $[A]$ and the orthogonal matrices Q_n satisfy the equation

$$[A][Q_n] = [Q_n][H_n] + [R], \tag{4.10}$$

where

$$[R] = h_{n+1,n}[q_{n+1}][e]_n^T \tag{4.11}$$

and the row vector $[e]_n^T$ is zero apart from its unit nth element [13].

The basic idea of the Arnoldi method is to use the eigenvectors and eigenvalues of the small Hessenberg matrix $[H_n]$ to find the eigenvectors and eigenvalues of the original matrix $[A]$. Substituting Eq. (4.8) into Eq. (4.10), we obtain

$$[D]^{1/2}[K]([L]-\sigma[D][K])^{-1}[D]^{1/2}[Q_n] = [Q_n][H_n] + [R]$$

or

$$([L]-\sigma[D][K])[K]^{-1}[D]^{-1/2}([Q_n][H_n] + [R]) = [D]^{1/2}[Q_n]. \tag{4.12}$$

We now suppose that s is an eigenvector of H_n corresponding to the eigenvalue λ and multiply s times Eq. (4.12) from the right to obtain

$$([L]-\sigma[D][K])[K]^{-1}[D]^{-1/2}([\lambda Q_n]s + [R]s) = [D]^{1/2}[Q_n]s$$

or

$$\begin{aligned}&([L]-\sigma[D][K])(\lambda[K]^{-1}[D]^{-1/2}[Q_n]s + [K]^{-1}[D]^{-1/2}[R]s)\\ &= [D]^{1/2}\left([D]^{1/2}[K][K]^{-1}[D]^{-1/2}\right)[Q_n]s.\end{aligned} \tag{4.13}$$

Eq. (4.13) can be simplified by making the following definitions

$$t = [K]^{-1}[D]^{-1/2}[Q_n]s, \tag{4.14}$$

$$r = [K]^{-1}[D]^{-1/2}[R]s. \tag{4.15}$$

With these definitions, Eq. (4.13) becomes

$$([L]-\sigma[D][K])(\lambda t + r) = [D][K]t.$$

Dividing the above equation by λ and rearranging terms, we finally obtain

$$([L]-(\sigma+1/\lambda)[D][K])(t + r/\lambda) = -[D]^{1/2}[R]s/\lambda^2. \tag{4.16}$$

https://doi.org/10.4208/cicp.101114.170615a Published online by Cambridge University Press

We may thus identify $t+r/\lambda$ as an eigenvector and $\sigma+1/\lambda$ as the corresponding eigenvalue. Since the eigenvector $t+r/\lambda$ is not normalized, an appropriate convergence criterion is

$$\frac{\| [D]^{1/2}[R]s \|}{|\lambda|^2 \| t+r/\lambda \|} \leq \text{tolerance}.$$

We have found the eigenvalues and eigenvectors of the helium ion He^+ in spheroidal coordinates using domain decomposition and the Arnoldi method and also using finite-differences of the derivatives together with the successive over-relaxation (SOR) method. For the results obtained by the SOR method, the eigenvalue was obtained using the Rayleigh quotient

$$\epsilon = \frac{(u, \mathbf{L}u)}{(u, \mathbf{D}u)}. \tag{4.17}$$

We first give a summary of the computer times for doing Arnoldi iterations. These calculations were performed with the Quad-Core Opteron computer at Nicholas Copernicus University in Torun, Poland.

The most time consuming step in the inverse iteration and inverse Arnoldi procedures is multiplying an inverse operator times a vector. Multiplying the inverse of a differential operator $\mathbf{L}$ times a vector $\mathbf{x}$ to obtain a vector $\mathbf{v}$ is equivalent to solving the partial differential equation $\mathbf{L}v = \mathbf{x}$. For solving the eigenvalue problem, our domain decomposition program thus becomes a subroutine for evaluating the product of an inverse operator times a vector. For this reason, the times given for a single Arnoldi iteration in Table 4 are similar to the times given for solving a PDE in Table 3. The main reason for the differences between the times given in the two tables is that the numerical results in

Table 4: CPU times (in seconds) for different numbers of intervals and different partitions of the entire domain. As in Tables 2 and 3, n_1 and n_2 give the number of intervals in the ν- and μ-direction. Column two gives the number of subregions into which the entire domain is divided. Column three gives the time taken to form and factor the interface matrix, and the last column gives the times for four/one Arnoldi iterations.

$n1 \times n2$	Partition	FFIM	AI
48×48	6×6	1.62	0.41/0.10
48×48	12×12	3.33	0.33/0.08
48×48	24×24	20.78	0.62/0.16
96×96	6×6	18.82	1.78/0.45
96×96	12×12	23.40	1.57/0.39
96×96	24×24	150.27	2.45/0.61
192×192	6×6	354.15	12.20/3.05
192×192	12×12	206.69	8.19/2.05
192×192	24×24	1212.05	11.08/2.78

https://doi.org/10.4208/cicp.101114.170615a Published online by Cambridge University Press

Table 3 were produced using the 8-core Xeon computer, while the numerical results in Table 4 were produced using the Quad-Core Opteron computer.

Notice that for the 96×96 and 192×192 grids the times given for factoring the interface matrix and performing a single Arnoldi iteration – once the interface matrix is factored – are comparable or smaller for a 12×12 partition of the region than for a 6×6 partition of the region. The reason for this is that the length of time necessary for solving a set of linear equations using Gaussian elimination goes as N^3 where N is the number of grid points. Even though a finer partition of the region has more subregions, the time for solving the set of linear equations in each subregion is very much smaller. For grids with a large number of points, a finer partition of the domain is generally more efficient.

We would like now to compare the accuracy of the domain decomposition and SOR methods. As we have said before the accuracy of solutions of PDEs using third-order Hermite splines goes as h^4 and is independent of how the region is partitioned into subregions. As for the SOR calculations, they were performed for the region as a whole. We may thus compare the numerical accuracy of the spline calculations of the energy and wave functions without considering how the entire region was divided into subregions. The accuracy of the Arnoldi calculations using third-order splines and domain decomposition and the SOR methods are compared in Table 5.

Table 5: The difference between the calculated energies and their analytic values ΔE and the corresponding differences for the wave function Δu. As in the previous tables, n_1 and n_2 give the number of intervals in the ν- and μ-directions.

$n1\times n2$	24×24	48×48	96×96	192×192
		ΔE		
Arnoldi	2×10^{-5}	1×10^{-6}	6×10^{-8}	4×10^{-9}
SOR	2×10^{-5}	1×10^{-6}	5×10^{-8}	4×10^{-9}
	1×10^{-7}	5×10^{-9}	2×10^{-11}	4×10^{-13}
		Δu		
Arnoldi	1×10^{-4}	1×10^{-5}	8×10^{-7}	5×10^{-8}
SOR	3×10^{-4}	2×10^{-6}	8×10^{-8}	2×10^{-8}
	4×10^{-5}	4×10^{-8}	2×10^{-10}	1×10^{-12}

The spline calculations using the Arnoldi method and the SOR calculations using finite difference approximations of the derivative are both very accurate. They produce energies at the micro-Hartree level. However, the SOR method produces the same level of accuracy with fewer points. The first line of entries for the energy (ΔE) and wave function (Δu) for the SOR method in Table 5 were performed using a sufficient number of SOR iterations to produce the same level of accuracy for the energy as for the Arnoldi method. The SOR iterations were then continued to obtain the optimum results for the SOR method shown in the second line of SOR entries. Notice that the spline-Arnoldi

https://doi.org/10.4208/cicp.101114.170615a Published online by Cambridge University Press

values for a grid with 192×192 points has the same level of accuracy as SOR calculations with 48×48. The reason for the higher level of accuracy of the SOR results is that they were obtained using an eighth-order finite difference formula, while the Arnoldi results were obtained using third-order splines. The level of accuracy of the SOR values of the energy and the wave function given in this table are beyond the level of accuracy of the Hartree-Fock model itself; however, such accurate values can be used as a reference for basis-set calculations. Within recent years, Kobus, Moncrieff and Wilson have performed a number of very accurate benchmark calculations of this kind [14–16].

The computer times required for solving the model problem for different numbers of grid points are given in Table 6. As for the results shown in Tables 4 and 5, these calculations were performed with the Quad-Core Opteron computer.

Table 6: The number of SOR iterations and the corresponding CPU times in seconds for calculations with different numbers of intervals. The integers n_1 and n_2 give the number of intervals in the ν- and μ-direction. The first row of entries gives the number of SOR iterations and time for doing calculations at the same level of accuracy as for the Arnoldi method, while the second row of entries gives the number of SOR iteration and times for an optimal SOR calculation which pushes the calculation to the limit of its accuracy.

$n_1\times n_2$	24×24	48×48	96×96	192×192
	$220/5.0\times10^{-3}$	$290/2.4\times10^{-2}$	$480/1.6\times10^{-1}$	$1240/1.7\times10^{0}$
	$230/7.0\times10^{-3}$	$330/3.9\times10^{-2}$	$590/2.1\times10^{-1}$	$1700/2.2\times10^{0}$

These times compare favorably with the times for doing calculations using Hermite splines and the domain decomposition method. We note the number of SOR iterations becomes much larger for calculations for grids with very many points and a corresponding higher-order of accuracy.

Although the domain decomposition and SOR methods have been described in this article as two alternate approaches, the two methods can very well be used in conjunction with each other. One can use domain decomposition to partition a region into subregions and then use the SOR method or another iterative method to solve the PDE in each subregion. As noted in the first article of this series, the preconditioned conjugate gradient method is very efficient in the broad interior of the region where the differential operator $\mathbf{L}$ is positive definite but cannot be used for the entire region because the coefficients of the derivatives go to zero on the boundary. Gaussian elimination and the SOR method can be used in all subregions.

5 Conclusion

The numerical results in this article show that very accurate calculations of molecular properties can be obtained numerically using either the spline collocation method with domain decomposition or with the SOR method with finite difference approximations of the derivatives. Given enough grid points either method can attain a desired level of

https://doi.org/10.4208/cicp.101114.170615a Published online by Cambridge University Press

accuracy within the double precision floating point arithmetic. Due to the higher-order difference formula used with the SOR method, this method produces the same level of accuracy as the spline collocation method with fewer numbers of points. The domain decomposition method has the strength that it can be applied to problems with a large number of grid points. The time required to solve a PDE for a fine grid with a large number of points goes down as the number of partitions increases. Indeed, the technique of dividing the entire region into an arbitrary number of subregions by vertical and horizontal surfaces perpendicular to the coordinate axis was developed with the two-electron pair equation in mind. The pair equation for diatomic molecules has five independent variables corresponding to the ν and μ variables of each electron and to an angle ϕ, which is the difference of the azimuthal angles ϕ_1 and ϕ_2. A grid with 48 intervals for each of the variables will result in matrices with more than eight billions rows and columns. It is to this fundamental correlation problem we now turn our attention. By solving the two-electron pair equation, we will be able to perform the same kinds of numerical coupled-cluster calculations for diatomic molecules as has been carried out for atoms [17].

References

[1] C. F. Fischer, Comp. Phys. Rep. 3 (1986) 273.
[2] G. Gaigalas and C. F. Fischer, Comp. Phys. Commun. 98 (1996) 255.
[3] E. A. McCullough Jr., Comp. Phys. Rep. 4 (1986) 265.
[4] D. Heinemann, D. Kolb, B. Fricke, Chem. Phys. Lett. 137 (1987) 180-182
[5] D. Heinemann, B. Fricke, D. Kolb, Phys. Rev. A, 38 (1988) 4994-5001
[6] L. Laaksonen, P. Pyykkö, and D. Sundholm, Comp. Phys. Rep. 4 (1986) 313.
[7] J. Kobus, L. Laaksonen, and D. Sundholm, Comput. Phys. Commun. 98 (1996) 346.
[8] J. Kobus, Comp. Phys. Commun. 184 (2013) 799.
[9] J. C. Morrison *et al.*, Commun. Comput. Phys. 5 (2009) 959.
[10] B. Bialecki and G. Fairweather, SIAM J. Sci. Comput., 16 (1995) 330.
[11] B. Bialecki, SIAM J. Numer. Anal. 35 (1998) 617.
[12] M. T. Heath, Scientific Computing: An Introductory Survey, McGraw-Hill Higher Education, 1996.
[13] L. N. Trefethen and I. D. Bau, Numerical Linear Algebra, SIAM, Philadelphia, 1997.
[14] J. Kobus, D. Moncrieff, and S. Wilson, Phys. Rev. A 62 (2000) 062503/1.
[15] J. Kobus, D. Moncrieff, and S. Wilson, Journal of Physics B: Atomic Molecular and Optical Physics 40 (2007) 877.
[16] J. Kobus, Phys. Rev. A 91 (2015) 022501.
[17] I. Lingren and J. Morrison, Atomic Many-Body Theory, Springer Series on Atoms and Molecules, vol. 3, Springer-Verlag, Berlin, 1986.

https://doi.org/10.4208/cicp.101114.170615a Published online by Cambridge University Press