SMAI-JCM

SMAI Journal of Computational Mathematics

Domain decomposition algorithms for the two dimensional nonlinear Schrödinger equation and simulation of Bose–Einstein condensates

Christophe Besse & Feng Xing

Volume 2 (2016), p. 277-300.

<http://smai-jcm.cedram.org/item?id=SMAI-JCM_2016__2__277_0>

SMAI Journal of Computational Mathematics
Vol. 2, 277-300 (2016)

Domain decomposition algorithms for the two dimensional nonlinear Schrödinger equation and simulation of Bose–Einstein condensates

Christophe Besse [1]
Feng Xing [2]

[1] Institut de Mathématiques de Toulouse UMR5219, Université de Toulouse; CNRS, UPS IMT, F-31062 Toulouse Cedex 9, France
E-mail address: christophe.besse@math.univ-toulouse.fr
[2] Laboratoire de Mathématiques J.A. Dieudonné, UMR 7351 CNRS, University Nice Sophia Antipolis, team COFFEE, INRIA Sophia Antipolis Méditerranée, Parc Valrose 06108 Nice Cedex 02, France, and BRGM Orlï¿½ans France
E-mail address: feng.xing@unice.fr.

Abstract. In this paper, we apply the optimized Schwarz method to the two dimensional nonlinear Schrödinger equation and extend this method to the simulation of Bose–Einstein condensates (Gross–Pitaevskii equation). We propose an extended version of the Schwartz method by introducing a preconditioned algorithm. The two algorithms are studied numerically. The experiments show that the preconditioned algorithm improves the convergence rate and reduces the computation time. In addition, the classical Robin condition and a newly constructed absorbing condition are used as transmission conditions.

Math. classification. 35Q55, 65M55, 65Y05, 65M60.

Keywords. nonlinear Schrödinger equation, rotating Bose–Einstein condensate, optimized Schwarz method, preconditioned algorithm, parallel algorithm.

1. Introduction

We are interested in solving the nonlinear Schrödinger equation and the Gross–Pitaevskii (GPE) equation by the optimized Schwarz method. A large number of articles and books [19, 17, 16, 18] are devoted to this method for different kinds of equations, for example the Poisson equation [22], the Helmholtz equation [14, 20] and the convection-diffusion equation [25]. Recently, the authors of [21, 5, 13, 12] applied the domain decomposition method to the linear or nonlinear Schrödinger equation. More specificaly, in [13, 12], the authors proposed some newly efficient and scalable Schwarz algorithms for 1d or 2d linear Schrödinger equation and for 1d nonlinear Schrödinger equation. These new algorithms could ensure high scalability and reduce computation time. In this paper, we extend these works to the two dimensional nonlinear case.

The nonlinear Schrödinger equation defined on a two dimensional bounded spatial domain $\Omega := (x_l, x_r) \times (y_b, y_u)$, $x_l, x_r, y_b, y_u \in \mathbb{R}$ and $t \in (0, T)$ with general real potential $V(t, x, y) + f(\cdot)$ reads

$$\begin{cases} i\partial_t u + \Delta u + V(t,x,y)u + f(u)u = 0, \ (t,x,y) \in (0,T) \times \Omega, \\ u(0,x,y) = u_0(x,y), \end{cases} \tag{1.1}$$

where $u_0 \in L^2(\Omega)$ is the initial datum. We complement the equation with homogeneous Neumann boundary condition on bottom and top boundaries and Fourier–Robin boundary conditions on left and right boundaries. They read:

$$\partial_{\mathbf{n}} u = 0, \ (x,y) \in (x_l, x_r) \times \{y_b, y_u\}, \quad \partial_{\mathbf{n}} u + Su = 0, \ (x,y) \in \{x_l, x_r\} \times (y_b, y_u), \tag{1.2}$$

where $\partial_{\mathbf{n}}$ denotes the normal directive, $\mathbf{n}$ being the outwardly unit vector on the boundary $\partial\Omega$ (see Figure 1.1). The operator S is given by

$$Su = -ip \cdot u,\ p \in \mathbb{R}^+, \tag{1.3}$$

$$\text{or } Su = -i\sqrt{i\partial_t + \Delta_\Gamma + V + f(u)}u, \tag{1.4}$$

where $\Gamma = \{x_l, x_r\} \times (y_b, y_u)$. The Laplace–Beltrami operator Δ_Γ is ∂_y^2 in our case.

The operator S in (1.4) is a pseudo differential operator constructed recently in [2] as an absorbing boundary operator, which is used to approximate the exact solution of the problem defined on $\mathbb{R}^2$, restricted to a bounded space domain.

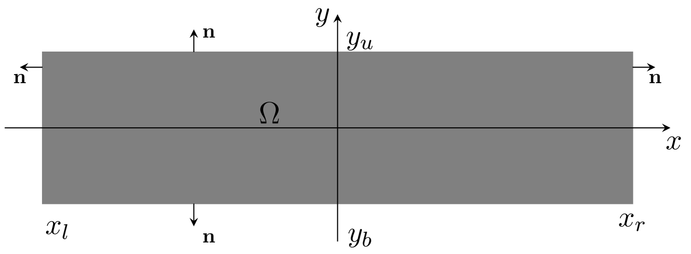


Figure 1.1. Spatial bounded domain $\Omega = (x_l, x_r) \times (y_b, y_u)$.

Recently, the Schwarz algorithms have been applied to the one dimensional linear or nonlinear Schrödinger equation [13]. If the potential is linear and independent of time, then an interface problem allows to construct a global in space operator. It is possible to assemble it in parallel without too much computational efforts. Thanks to this operator, a new algorithm was introduced which is mathematically equivalent to the original Schwarz algorithm, but requires less computation time. If the potential is general, the authors used a pre-constructed linear operator as preconditioner, which leads to a preconditioned algorithm. The preconditioner allows to reduce both the number of iterations and the computation time. These new algorithms have been extended to the two dimensional linear Schrödinger equation [12], which also shows the effectiveness of the new algorithms. In this article, we propose to extend the results to the 2d nonlinear case and to the simulation of Bose–Einstein condensates. Following the naming in [13], we refer to the algorithm given by (2.2) as "the classical algorithm" and to the algorithm that will be presented in Section 3 as "the preconditioned algorithm".

The paper is organized as follows. We present in Section 2 the classical algorithm and some details about the discretization. A preconditioned algorithm is proposed in Section 3. In Section 4, we show the implementation of these algorithms on parallel computers. Numerical experiments are performed in Section 5 and we focused on simulation of Bose–Einstein condensates there. The last section draws a conclusion and suggests some future directions of research.

2. Classical algorithm

2.1. Classical optimized Schwarz algorithm

Let us discretize uniformly with N_T intervals the time domain $(0, T)$. We define $\Delta t = T/N_T$ to be the time step. The usual semi-discrete in time scheme developed by Durán and Sanz–Serna [15] applied to (1.1) reads as

$$i\frac{u_n - u_{n-1}}{\Delta t} + \Delta\frac{u_n + u_{n-1}}{2} + \frac{V_n + V_{n-1}}{2}\frac{u_n + u_{n-1}}{2} + f(\frac{u_n + u_{n-1}}{2})\frac{u_n + u_{n-1}}{2} = 0, \quad 1 \leqslant n \leqslant N_T$$

where $u_n(x,y),(x,y)\in\Omega$ denotes the approximation of the solution $u(t_n,x,y)$ to the Schrödinger equation (1.1) at time $t_n=n\Delta t$ and $V_n(x,y)=V(t_n,x,y)$. By introducing new variables $v_n=(u_n+u_{n-1})/2$ with $v_0=u_0$ and $W_n=(V_n+V_{n-1})/2$, this scheme can be written as

$$\mathscr{L}_{\mathbf{x}}v_n=2i\frac{u_{n-1}}{\Delta t},\tag{2.1}$$

where the operator $\mathscr{L}_{\mathbf{x}}$ is defined by

$$\mathscr{L}_{\mathbf{x}}v_n:=\frac{2i}{\Delta t}v_n+\Delta v_n+W_nv_n+f(v_n)v_n.$$

For any $1\leqslant n\leqslant N_T$, the equation (2.1) is stationary. We can therefore apply the optimized Schwarz method. Let us decompose the spatial domain Ω into N subdomains $\Omega_j=(a_j,b_j), j=1,2,...,N$ without overlap as shown in Figure 2.1 for $N=3$. The Schwarz algorithm is an iterative process and we identify the iteration number thanks to label k. We denote by $v_{j,n}^k$ the solution on subdomain Ω_j at iteration $k=1,2,...$ of the Schwarz algorithm (resp $u_{j,n}^k$). Assuming that $u_{0,n-1}$ is known, the optimized Schwarz algorithm for (2.1) consists in applying the following sequence of iterations for $j=1,2,...,N$

$$\begin{cases}\mathscr{L}_{\mathbf{x}}v_{j,n}^k=\dfrac{2i}{\Delta t}u_{j,n-1},\ (x,y)\in\Omega_j,\\ \partial_{\mathbf{n}_j}v_{j,n}^k+\overline{S}_jv_{j,n}^k=\partial_{\mathbf{n}_j}v_{j-1,n}^{k-1}+\overline{S}_jv_{j-1,n}^{k-1},\ x=a_j,\ y\in(y_b,y_u),\\ \partial_{\mathbf{n}_j}v_{j,n}^k+\overline{S}_jv_{j,n}^k=\partial_{\mathbf{n}_j}v_{j+1,n}^{k-1}+\overline{S}_jv_{j+1,n}^{k-1},\ x=b_j,\ y\in(y_b,y_u),\end{cases}\tag{2.2}$$

with a special treatment for the two extreme subdomains Ω_1 and Ω_N since the boundary conditions are imposed on $\{x=a_1\}\times(y_b,y_u)$ and $\{x=b_N\}\times(y_b,y_u)$

$$\partial_{\mathbf{n}_1}v_{1,n}^k+\overline{S}_jv_{1,n}^k=0,x=a_1,\quad \partial_{\mathbf{n}_N}v_{N,n}^k+\overline{S}_jv_{N,n}^k=0,\ x=b_N.$$

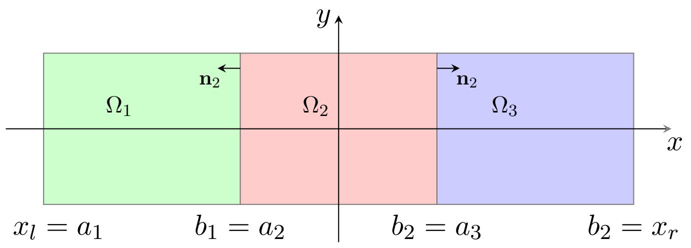


Figure 2.1. Domain decomposition without overlap, $N=3$.

Various transmission operator $\overline{S}_j$ can be considered, such as the classical widely used Robin transmission condition

$$\text{Robin}:\quad \overline{S}_jv=-ip\cdot v,\ p\in\mathbb{R}^+.\tag{2.3}$$

Traditionally, the optimal transmission operator is given in term of transparent boundary conditions (TBCs). For the nonlinear two dimensional Schrödinger equation, we only have access to approximated version of the TBCs given by the recently constructed absorbing boundary condition S_{pade}^m [2, 3] which is used as the transmission condition here

$$S_{\text{pade}}^m:\quad \overline{S}_jv=-i\sum_{s=0}^m a_s^mv+i\sum_{s=1}^m a_s^md_s^m\varphi_{j,s},\ x=a_j,b_j.\tag{2.4}$$

The auxiliary functions $\varphi_{j,s}$, $j=1,2,...,N$,$s=1,2,...,m$ are defined as solution of the set of equations

$$\Big(\frac{2i}{\Delta t}+\Delta_{\Gamma_j}+W+f(v)+d_s^m\Big)\varphi_{j,s}(x,y)=v,\ (x,y)\in(a_j,b_j)\times(y_b,y_u).\tag{2.5}$$

The operator S^m_{pade} [2, 3] is originally constructed by using some pseudo differential techniques. Numerically it is approximated by the Padé approximation of order m

$$S^m_{\text{pade}}v = -i\sqrt{\frac{2i}{\Delta t} + \Delta_{\Gamma_j} + W + f(v)v} \approx \Big(-i\sum_{s=0}^{m} a^m_s + i\sum_{s=1}^{m} a^m_s d^m_s (\frac{2i}{\Delta t} + \Delta_{\Gamma_j} + W + f(v) + d^m_s)^{-1}\Big)v,$$

where $\Gamma_j = \{x = a_j, b_j\} \times (y_b, y_u)$. The Laplace–Beltrami operator Δ_{Γ_j} is ∂_{yy} in our case and the constant coefficients are $a^m_s = e^{i\theta/2}/(m\cos^2(\frac{(2s-1)\pi}{4m}))$, $d^m_s = e^{i\theta}\tan^2(\frac{(2s-1)\pi}{4m})$, $s = 0, 1, ..., m$, $\theta = \frac{\pi}{4}$.

Let us introduce the fluxes $l^k_{j,n}$ and $r^k_{j,n}$ defined as

$$\begin{aligned} l^k_{j,n}(y) &= \partial_{\mathbf{n}_j} v^k_{j,n}(a_j, y) + \overline{S}_j v^k_{j,n}(a_j, y),\ y \in (y_b, y_u), \\ r^k_{j,n}(y) &= \partial_{\mathbf{n}_j} v^k_{j,n}(b_j, y) + \overline{S}_j v^k_{j,n}(b_j, y),\ y \in (y_b, y_u), \end{aligned}$$

with a special definition for the two extreme subdomains: $l^k_{1,n} = r^k_{N,n} = 0$. Thus, the algorithm (2.2) can be splitted into local problems on subdomains $\Omega_j, j = 1, 2, ..., N$

$$\begin{cases} \mathscr{L}_{\mathbf{x}} v^k_{j,n} = \dfrac{2i}{\Delta t} u_{j,n-1}, \\ \partial_{\mathbf{n}_j} v^k_{j,n} + \overline{S}_j v^k_{j,n} = l^k_{j,n},\ x = a_j, \\ \partial_{\mathbf{n}_j} v^k_{j,n} + \overline{S}_j v^k_{j,n} = r^k_{j,n},\ x = b_j, \end{cases} \tag{2.6}$$

and flux problems

$$\begin{cases} l^{k+1}_{j,n} = -r^k_{j-1,n} + 2\overline{S}_j v^k_{j-1,n}(b_{j-1}, y),\ j = 2, 3, ..., N, \\ r^{k+1}_{j,n} = -l^k_{j+1,n} + 2\overline{S}_j v^k_{j+1,n}(a_{j+1}, y),\ j = 1, 2, ..., N-1. \end{cases} \tag{2.7}$$

Remark 2.1. Let us remark that the transmission operator (1.4) is singular at the corner of the subdomains in the continuous case. Thus, we consider mostly its discrete form in this paper. In the numerical tests, we will use a large domain such that the solution on the boundary $\partial\Omega$ are sufficiently small below the computer precision.

Remark 2.2. There are obviously other ways to decompose the complete domain. One way is illustrated in Figure 2.2 (left) for $N = 4$. The intervals (x_l, x_r) and (y_b, y_u) are simultaneously decomposed into subintervals in both spatial directions. In this configuration, an artificial cross point appears. It is well known that the domain decomposition method with cross points is a difficult problem since the problem becomes singular at this point [23]. Another possibility is illustrated in Figure 2.2 (right) for $N = 3$. The entire domain is decomposed into an ellipsis and rings. However, this approach has many disadvantages for parallel computing. Indeed, we would like to control the number of cells for the meshes of each subdomains. Their sizes have to be equivalent to insure a good balance between different processes. Thus, we restrict ourselves in this paper to the first description (see Figure 2.1).

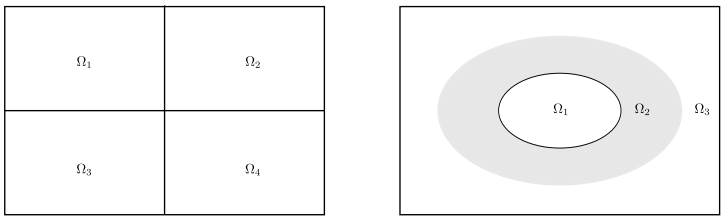


Figure 2.2. Two other ways of domain decomposition.

2.2. Preliminaries related to space discretization

Let $(a,b)\times(y_b,y_u), a,b\in\mathbb{R}$ be a bounded domain, which can be the complete domain Ω or any sub domain $\Omega_j, j=1,2,...N$. Without loss of generality, we present the space discretization of the semi-discrete Schrödinger equation defined on

$$\begin{cases} \dfrac{2i}{\Delta t}v+\Delta v+W(x,y)v+f(v)v=\dfrac{2i}{\Delta t}h(x,y), \\ \partial_{\mathbf{n}}v+\overline{S}v=l(x,y),\ x=a,\ y\in(y_b,y_u), \\ \partial_{\mathbf{n}}v+\overline{S}v=r(x,y),\ x=b,\ y\in(y_b,y_u), \\ \partial_{\mathbf{n}}v=0,\ y=y_b,\ y=y_u, \end{cases} \tag{2.8}$$

where $W(x,y)v+f(v)v$ plays the role of the semi-discrete potential in (2.1) and $l(x,y)$, $r(x,y)$ are two functions. The operator $\overline{S}$ is Robin or S^m_{pade} given respectively by (2.3) and (2.4). The discretization with S^m_{pade} is initially proposed in [2, 3]. The stability of the scheme is observed according to the numerical results. However, theoretical theory of stability is still missing.

If $f\neq 0$, then the system (2.8) is nonlinear. The computation of v is accomplished by a fixed point procedure. If we consider the Robin transmission condition, we take $\zeta^0=h$ and compute the solution v as the limit of the iterative procedure with respect to the label q, $q=1,2,...$

$$\begin{cases} \Big(\dfrac{2i}{\Delta t}+\Delta+W+f(\zeta^{q-1})\Big)\zeta^q=\dfrac{2i}{\Delta t}h, \\ \partial_{\mathbf{n}}\zeta^q-ip\cdot\zeta^q=l,\ x=a, \\ \partial_{\mathbf{n}}\zeta^q-ip\cdot\zeta^q=r,\ x=b. \end{cases} \tag{2.9}$$

For the transmission condition S^m_{pade}, we take $\zeta^0=h$ and $\phi^0_s=0$, $s=1,2,...,m$. The unknowns v and $\varphi_s, s=1,2,...,m$ are computed as the limit (with respect to q) of ζ^q and ϕ^q_s, $s=1,2,\cdots,m$, which are solutions of the following coupled system

$$\begin{cases} (\dfrac{2i}{\Delta t}+\Delta+W_n+f(\zeta^{q-1})\zeta^q=\dfrac{2i}{\Delta t}h, \\ (\dfrac{2i}{\Delta t}+W+\Delta_\Gamma+d^m_s)\phi^q_s=\zeta^q-f(\zeta^{q-1})\phi^{q-1}_s, \\ \partial_{\mathbf{n}}\zeta^q-i\displaystyle\sum_{s=0}^m a^m_s\zeta^q+i\sum_{s=1}^m a^m_s d^m_s\phi^q_s=l,\ x=a, \\ \partial_{\mathbf{n}}\zeta^q-i\displaystyle\sum_{s=0}^m a^m_s\zeta^q+i\sum_{s=1}^m a^m_s d^m_s\phi^q_s=r,\ x=b. \end{cases} \tag{2.10}$$

The spatial approximation is realized with the standard $\mathbb{Q}_1$ finite element method with uniform mesh. The mesh size of a discrete element is $(\Delta x,\Delta y)$. We denote by N_x (resp. N_y) the number of nodes in x (resp. y) direction on each subdomain. Let us denote by $\mathbf{v}$ (resp. $\mathbf{h}$) the nodal interpolation vector of v (resp. h), $\boldsymbol{\zeta}^q$ the nodal interpolation vector of ζ^q, $\mathbf{l}$ (resp. $\mathbf{r}$) the nodal interpolation vector of $l(x,y)$ (resp. $r(x,y)$), $\mathbb{M}$ the mass matrix, $\mathbb{S}$ the stiffness matrix and $\mathbb{M}_W$ the generalized mass matrix with respect to $\int_\Omega Wv\phi dx$. Let $\mathbb{M}^\Gamma$ the boundary mass matrix, $\mathbb{S}^\Gamma$ the boundary stiffness matrix and $\mathbb{M}^\Gamma_W$ (resp. $\mathbb{M}^\Gamma_W$) the generalized boundary mass matrix with respect to $\int_\Gamma Wv\phi d\Gamma$ (resp. $\int_\Gamma f(v)\phi d\Gamma$). We denote by Q_l (resp. Q_r) the restriction operators (matrix) from Ω to $\{a\}\times(y_b,y_u)$ (resp. $\{b\}\times(y_b,y_u)$) and $Q^\top=(Q^\top_l,Q^\top_r)$ where "$\cdot^\top$" is the standard notation of the transpose of a matrix or a vector. The matrix formulation for (2.9) is therefore given by

$$\Big(\mathbb{A}+ip\cdot\mathbb{M}^\Gamma\Big)\boldsymbol{\zeta}^q=\frac{2i}{\Delta t}\mathbb{M}\mathbf{h}-\mathbb{M}^\Gamma_{f(\boldsymbol{\zeta}^{q-1})}\boldsymbol{\zeta}^{q-1}-\mathbb{M}^\Gamma Q^\top\begin{pmatrix}\mathbf{l}\\ \mathbf{r}\end{pmatrix}, \tag{2.11}$$

where $\mathbb{A} = \frac{2i}{\Delta t}\mathbb{M} - \mathbb{S} + \mathbb{M}_W$. The size of this linear system is $N_x \times N_y$. If we consider the transmission condition S^m_{pade}, we have

$$\begin{aligned}
\mathcal{A}\begin{pmatrix}\boldsymbol{\zeta}^q\\ \boldsymbol{\phi}_1^q\\ \boldsymbol{\phi}_2^q\\ \vdots\\ \boldsymbol{\phi}_m^q\end{pmatrix} &:= \begin{pmatrix}\mathbb{A} + i(\sum_{s=1}^m a_s^m)\cdot\mathbb{M}^\Gamma & \mathbb{B}_1 & \mathbb{B}_2 & \cdots & \mathbb{B}_m\\ \mathbb{C} & \mathbb{D}_1 & & & \\ \mathbb{C} & & \mathbb{D}_2 & & \\ \vdots & & & \ddots & \\ \mathbb{C} & & & & \mathbb{D}_m\end{pmatrix}\begin{pmatrix}\boldsymbol{\zeta}^q\\ \boldsymbol{\phi}_1^q\\ \boldsymbol{\phi}_2^q\\ \vdots\\ \boldsymbol{\phi}_m^q\end{pmatrix}\\
&= \frac{2i}{\Delta t}\begin{pmatrix}\mathbb{M}\mathbf{h}\\ \mathbf{0}\\ \mathbf{0}\\ \vdots\\ \mathbf{0}\end{pmatrix} - \begin{pmatrix}\mathbb{M}_{f(\boldsymbol{\zeta}^{q-1})}\boldsymbol{\zeta}^{q-1}\\ \mathbb{M}^\Gamma_{f(\boldsymbol{\zeta}^{q-1})}\boldsymbol{\phi}_1^{q-1}\\ \vdots\\ \mathbb{M}^\Gamma_{f(\boldsymbol{\zeta}_j^{q-1})}\boldsymbol{\phi}_m^{q-1}\end{pmatrix} - \begin{pmatrix}\mathbb{M}^\Gamma\cdot Q^\top\begin{pmatrix}\mathbf{l}\\ \mathbf{r}\end{pmatrix}\\ 0\\ \vdots\\ 0\end{pmatrix}.
\end{aligned} \tag{2.12}$$

with

$$\begin{aligned}
\mathbb{B}_s &= -ia_s^m d_s^m \mathbb{M}^\Gamma Q^\top, \quad 1 \leqslant s \leqslant m,\\
\mathbb{C} &= -Q\mathbb{M}^\Gamma,\\
\mathbb{D}_s &= Q(\frac{2i}{\Delta t}\mathbb{M}^\Gamma - \mathbb{S}^\Gamma + \mathbb{M}^\Gamma_W + d_s^m\mathbb{M}^\Gamma)Q^\top, \quad 1 \leqslant s \leqslant m.
\end{aligned}$$

It is a linear system with unknown $(\boldsymbol{\zeta}^q, \boldsymbol{\phi}_1^q, ..., \boldsymbol{\phi}_m^q)$ where $\boldsymbol{\phi}_s^q$ is the nodal interpolation of ϕ_s^q on the boundary and $\boldsymbol{\varphi}_s$ is the nodal interpolation of φ_s. The vectors $\mathbf{v}$ and $\boldsymbol{\varphi}_s$ are computed by

$$\mathbf{v} = \lim_{q\to\infty}\boldsymbol{\zeta}^q, \quad \boldsymbol{\varphi}_s = \lim_{q\to\infty}\boldsymbol{\phi}_s^q, \ s = 1, 2, ..., m.$$

In addition, the discrete form of the transmission operator $\overline{S}$ is given by

$$\begin{aligned}
\text{Robin}: \quad & \mathbf{Sv} = -ip\cdot\mathbf{v}, \ p \in \mathbb{R}^+,\\
S^m_{\text{pade}}: \quad & \mathbf{Sv} = -i\sum_{s=0}^m a_s^m\mathbf{v} + i\sum_{s=1}^m a_s^m d_s^m\boldsymbol{\varphi}_s.
\end{aligned} \tag{2.13}$$

Remark 2.3. The S^m_{pade} transmission condition involves a larger linear system to solve than the one of the Robin transmission condition. The cost of the algorithm with the S^m_{pade} transmission condition is therefore more expensive.

If the potential is linear $f = 0$, then from the system (2.8) we have

$$\text{Robin}: \quad \Big(\mathbb{A} + ip\cdot\mathbb{M}^\Gamma\Big)\mathbf{v} = \frac{2i}{\Delta t}\mathbb{M}\mathbf{h} - \mathbb{M}^\Gamma Q^\top\begin{pmatrix}\mathbf{l}\\ \mathbf{r}\end{pmatrix}, \tag{2.14}$$

and

$$S_{\text{pade:}} \quad \mathcal{A}\begin{pmatrix}\mathbf{v}\\ \boldsymbol{\varphi}_1\\ \boldsymbol{\varphi}_2\\ \vdots\\ \boldsymbol{\varphi}_m\end{pmatrix} = \frac{2i}{\Delta t}\begin{pmatrix}\mathbb{M}\mathbf{h}\\ \mathbf{0}\\ \mathbf{0}\\ \vdots\\ \mathbf{0}\end{pmatrix} - \begin{pmatrix}\mathbb{M}^\Gamma\cdot Q^\top\begin{pmatrix}\mathbf{l}\\ \mathbf{r}\end{pmatrix}\\ 0\\ \vdots\\ 0\end{pmatrix}. \tag{2.15}$$

Directly from the definition of $\mathcal{A}$, (2.15) can be written as one equation for $\mathbf{v}$

$$\Big(\mathbb{A} + i(\sum_{s=1}^m a_s^m)\cdot\mathbb{M}^\Gamma - \sum_{s=1}\mathbb{B}_s\mathbb{D}_s^{-1}\mathbb{C}_s\Big)\mathbf{v} = \frac{2i}{\Delta t}\mathbb{M}\mathbf{h} - \mathbb{M}^\Gamma Q^\top\begin{pmatrix}\mathbf{l}\\ \mathbf{r}\end{pmatrix}. \tag{2.16}$$

Note that numerically, we implement (2.14) and (2.15) to compute $\mathbf{v}$ (and $\boldsymbol{\varphi}_s$, $s = 1, 2, ..., m$).

2.3. Classical discrete algorithm

Following what is done for a bounded domain in the previous subsection, we discretize the equations (2.6) and (2.7) on each subdomain Ω_j at each time step $n = 1, 2, ..., N_T$. Accordingly, on each subdomain Ω_j, let us denote by

- $\mathbb{A}_{j,n} = \frac{2i}{\Delta t}\mathbb{M}_j - \mathbb{S}_j + \mathbb{M}_{j,W_n}$ where $\mathbb{M}_j$ is the mass matrix, $\mathbb{S}_j$ is the stiffness matrix, $\mathbb{M}_{j,W_n}$ is the generalized mass matrix with respect to $\int_{\Omega_j} W_n v\phi dx$,
- $\mathbb{M}_{W_n}^{\Gamma_j}$ the generalized boundary mass matrix with respect to $\int_{\Gamma_j} W_n v\phi d\Gamma$, $\mathbb{M}_f^{\Gamma_j}$ the generalized boundary mass matrix with respect to $\int_{\Gamma_j} f(v)\phi d\Gamma$ where $\Gamma_j = \{x = a_j, b_j\} \times (y_b, y_u)$,
- $Q_{j,l}$ and $Q_{j,r}$ the restriction operators (matrix) from Ω_j to its boundary $\{a_j\} \times (y_b, y_u)$ and $\{b_j\} \times (y_b, y_u)$ respectively, $Q_j^\top = (Q_{j,l}^\top, Q_{j,r}^\top)$,
- $\mathbb{B}_{j,s}$, $\mathbb{C}_j$, $\mathbb{D}_{j,n,s}$ the matrix associated with the operator S_{pade}^m,
- $\mathbf{v}_{j,n}^k$ (resp. $\mathbf{u}_{j,n}^k$) the interpolation vectors of $v_{j,n}^k$ (resp. $u_{j,n}^k$).

We denote by $\mathbf{l}_{j,n}^k$ (resp. $\mathbf{r}_{j,n}^k$) the nodal interpolation vector of $l_{j,n}^k$ (resp. $r_{j,n}^k$). The classical algorithm is initialized by an initial guess of $\mathbf{l}_{j,n}^0$ and $\mathbf{r}_{j,n}^0$, $j = 1, 2, ..., N$. The boundary conditions for any subdomain Ω_j at iteration $k+1$ involve the knowledge of the values of the functions on adjacent subdomains Ω_{j-1} and Ω_{j+1} at prior iteration k. Thanks to the initial guess, we can *solve* the Schrödinger equation on each subdomain, allowing to build the new boundary conditions for the next step, *communicating* them to other subdomains. This procedure is summarized in (2.17) for $N = 3$ subdomains at iteration k.

$$\begin{pmatrix} \mathbf{r}_{1,n}^k \\ \mathbf{l}_{2,n}^k \\ \mathbf{r}_{2,n}^k \\ \mathbf{r}_{3,n}^k \end{pmatrix} \xrightarrow{Solve} \begin{pmatrix} \mathbf{v}_{1,n}^k \\ \mathbf{v}_{2,n}^k \\ \mathbf{v}_{3,n}^k \end{pmatrix} \to \begin{pmatrix} -\mathbf{r}_{1,n}^k + 2\mathbf{S}(Q_{1,r}\mathbf{v}_{1,n}^k) \\ -\mathbf{l}_{2,n}^k + 2\mathbf{S}(Q_{2,l}\mathbf{v}_{j,n}^k) \\ -\mathbf{r}_{2,n}^k + 2\mathbf{S}(Q_{2,r}\mathbf{v}_{j,n}^k) \\ -\mathbf{l}_{3,n}^k + 2\mathbf{S}(Q_{3,l}\mathbf{v}_{N,n}^k) \end{pmatrix} \xrightarrow{Comm.} \begin{pmatrix} \mathbf{r}_{1,n}^{k+1} \\ \mathbf{l}_{2,n}^{k+1} \\ \mathbf{r}_{2,n}^{k+1} \\ \mathbf{l}_{3,n}^{k+1} \end{pmatrix}. \tag{2.17}$$

Let us define the discrete interface vector by

$$\mathbf{g}_n^{k,\top} = (\mathbf{r}_{1,n}^{k,\top}, \cdots, \mathbf{l}_{j,n}^{k,\top}, \mathbf{r}_{j,n}^{k,\top}, \cdots, \mathbf{l}_{N,n}^{k,\top}).$$

Thanks to this definition, we give a new interpretation to the algorithm which can be written as

$$\mathbf{g}_n^{k+1} = \mathcal{R}_{h,n}\mathbf{g}_n^k = I - (I - \mathcal{R}_{h,n})\mathbf{g}_n^k. \tag{2.18}$$

where I is identity operator and $\mathcal{R}_{h,n}$ is an operator. The solution to this iteration process is given as the solution to the discrete interface problem

$$(I - \mathcal{R}_{h,n})\mathbf{g}_n = 0.$$

In conclusion, the classical algorithm is written as Algorithm 1.

Algorithm 1: Classical algorithm

```
1: Initialize the iteration by $\mathbf{g}_1^0$,
2: for $n = 1, 2, ..., N_T$ do
    2.1: while $||\mathbf{g}_n^{k+1} - \mathbf{g}_n^k|| > \varepsilon$, $\varepsilon \ll 1$ do
        $\mathbf{g}_n^{k+1} = \mathcal{R}_{h,n}\mathbf{g}_n^k$
    2.2: $\mathbf{g}_{n+1}^0 = \mathbf{g}_n^{k+1}$.
```

Remark 2.4. If $f = 0$, the discretization of (2.6) on each subdomain is

$$\text{Robin}: \quad \Big(\mathbb{A}_{j,n} + ip \cdot \mathbb{M}^{\Gamma_j}\Big)\mathbf{v}_{j,n}^k = \frac{2i}{\Delta t}\mathbb{M}\mathbf{u}_{j,n-1}^k - \mathbb{M}^{\Gamma_j} Q_j^\top \begin{pmatrix} \mathbf{l}_{j,n}^k \\ \mathbf{r}_{j,n}^k \end{pmatrix}, \tag{2.19}$$

$$S_{\text{pade}}: \quad \Big(\mathbb{A}_{j,n} + i(\sum_{s=1}^m a_s^m) \cdot \mathbb{M}^{\Gamma_j} - \sum_{s=1}^m \mathbb{B}_{j,s}\mathbb{D}_{j,n,s}^{-1}\mathbb{C}_j\Big)\mathbf{v}_{j,n}^k = \frac{2i}{\Delta t}\mathbb{M}\mathbf{u}_{j,n-1}^k - \mathbb{M}^{\Gamma_j} Q_j^\top \begin{pmatrix} \mathbf{l}_{j,n}^k \\ \mathbf{r}_{j,n}^k \end{pmatrix}. \tag{2.20}$$

3. Preconditioned algorithm

The application of the nonlinear operator $\mathcal{R}_{h,n}$ to $\mathbf{g}_n^k$ is expensive. In this section, we propose to add a preconditioner P^{-1} in (2.18), which leads to a preconditioned algorithm

$$\mathbf{g}_n^{k+1} = I - P^{-1}(I - \mathcal{R}_{h,n})\mathbf{g}_n^k. \tag{3.1}$$

Here P is a non singular matrix. To defined it, let us consider the free Schrödinger equation with a zero potential $V = 0$, $f = 0$. We show in Propositions 3.1 and 3.2 that in this case, the operator $\mathcal{R}_{h,n}$ is linear

$$\mathcal{R}_{h,n}\mathbf{g}_n^k = \mathcal{L}_h\mathbf{g}_n^k + \mathbf{d}_n, \tag{3.2}$$

where $\mathcal{L}_h$ is a block matrix as defined by (3.3) and $\mathbf{d}_n$ is a vector (the notation "MPI j" is used in the next section). The matrix $\mathcal{L}_h$ is independent of the time step n. The size of each block ,$X^{j,1}$, $X^{j,2}$, $X^{j,3}$, $X^{j,4}$ is $N_y \times N_y$.

$$\mathcal{L}_h = \left(\begin{array}{ccccccccc}
\overbrace{\quad}^{\text{MPI } 0} & \multicolumn{2}{c}{\overbrace{\qquad\qquad}^{\text{MPI } 1}} & \multicolumn{2}{c}{\overbrace{\qquad\qquad}^{\text{MPI } 2}} & & \multicolumn{2}{c}{\overbrace{\qquad\qquad\qquad}^{\text{MPI } N-2}} & \overbrace{\quad}^{\text{MPI } N-1} \\
 & X^{2,1} & X^{2,2} & & & & & & \\
X^{1,4} & & & & & & & & \\
 & & & X^{3,1} & X^{3,2} & & & & \\
 & X^{2,3} & X^{2,4} & & & & & & \\
 & & & & & \cdots & & & \\
 & & & X^{3,3} & X^{3,4} & & & & \\
 & & & & & & X^{N-1,1} & X^{N-1,2} & \\
 & & & & & \cdots & & & \\
 & & & & & & & & X^{N,1} \\
 & & & & & & X^{N-1,3} & X^{N-1,4} &
\end{array}\right). \tag{3.3}$$

Thus, we propose here

$$P = I - \mathcal{L}_h.$$

Note that since $\mathcal{L}_h$ is independent of time step n, the preconditioner is constructed once and used for all time steps. Thus, the preconditioned algorithm can be written with:

Algorithm 2: Preconditioned algorithm

```
1: Compute the preconditioner $P = I - \mathcal{L}_h$,
2: Initialize the iteration by $\mathbf{g}_1^0$,
3: for $n = 1, 2, ..., N_T$ do
    2.1: while $||\mathbf{g}_n^{k+1} - \mathbf{g}_n^k|| > \varepsilon$, $\varepsilon \ll 1$ do
        $\mathbf{g}_n^{k+1} = I - P^{-1}(I - \mathcal{R}_{h,n})\mathbf{g}_n^k$
    2.2: $\mathbf{g}_{n+1}^0 = \mathbf{g}_n^{k+1}$.
```

Proposition 3.1. *For the Robin transmission condition, assuming that $V = 0$ and $f = 0$, the matrix $\mathcal{L}_h$ takes the form* (3.3) *and $\mathcal{L}_h$ is independent of time step n. In addition, if the subdomains are equal, then*

$$\begin{aligned} &X^{2,1} = X^{3,1} = \cdots = X^{N,1},\ X^{2,2} = X^{3,2} = \cdots = X^{N-1,2},\\ &X^{2,3} = X^{3,3} = \cdots = X^{N-1,3},\ X^{1,4} = X^{2,4} = \cdots = X^{N-1,4}. \end{aligned} \tag{3.4}$$

Proof. First, by some straight forward calculations using (2.19) and (2.7), we can verify that

$$\begin{aligned} X^{j,1} &= -I - 2ip \cdot Q_{j,l}(\mathbb{A}_{j,n} + ip \cdot \mathbb{M}^{\Gamma_j})^{-1}\mathbb{M}^{\Gamma_j} Q_{j,l}^\top,\\ X^{j,2} &= -2ip \cdot Q_{j,l}(\mathbb{A}_{j,n} + ip \cdot \mathbb{M}^{\Gamma_j})^{-1}\mathbb{M}^{\Gamma_j} Q_{j,r}^\top,\\ X^{j,3} &= -2ip \cdot Q_{j,r}(\mathbb{A}_{j,n} + ip \cdot \mathbb{M}^{\Gamma_j})^{-1}\mathbb{M}^{\Gamma_j} Q_{j,l}^\top,\\ X^{j,4} &= -I - 2ip \cdot Q_{j,r}(\mathbb{A}_{j,n} + ip \cdot \mathbb{M}^{\Gamma_j})^{-1}\mathbb{M}^{\Gamma_j} Q_{j,r}^\top, \end{aligned} \tag{3.5}$$

and $\mathbf{d}_n^\top = (\mathbf{d}_{n,1,r}^\top, ..., \mathbf{d}_{n,j,l}^\top, \mathbf{d}_{n,j,r}^\top, ..., \mathbf{d}_{n,N,r}^\top)^\top$ with

$$\begin{aligned} \mathbf{d}_{n,j,l} &= 2ip \cdot Q_{j-1,r}(\mathbb{A}_{j-1,n} + ip \cdot \mathbb{M}^{\Gamma_{j-1}})\frac{2i}{\Delta t}\mathbf{u}_{j-1,n},\ j = 2, 3, ..., N,\\ \mathbf{d}_{n,j,r} &= 2ip \cdot Q_{j+1,l}(\mathbb{A}_{j+1,n} + ip \cdot \mathbb{M}^{\Gamma_{j+1}})^{-1}\frac{2i}{\Delta t}\mathbf{u}_{j+1,n},\ j = 1, 2, ..., N-1. \end{aligned} \tag{3.6}$$

Secondly, since $V = 0$, then

$$\mathbb{M}_{j,W_n} = 0, \quad \mathbb{A}_{j,1} = \mathbb{A}_{j,2} = ... = \mathbb{A}_{j,N_T} = \frac{2i}{\Delta t}\mathbb{M}_j - \mathbb{S}_j,\ j = 1, 2, ..., N. \tag{3.7}$$

Thus, the blocks $X^{j,1}$, $X^{j,2}$, $X^{j,3}$ and $X^{j,4}$ are both independent of time step n.
Finally, thanks to the hypothesis of the proposition, the geometry of each subdomain is identical. Thus, the various matrices coming from the assembly of the finite element methods are the same. Therefore, we have

$$\begin{aligned} \mathbb{M}_1 = \mathbb{M}_2 = ... = \mathbb{M}_N, \quad \mathbb{S}_1 = \mathbb{S}_2 = ... = \mathbb{S}_N, \quad \mathbb{M}^{\Gamma_1} = \mathbb{M}^{\Gamma_2} = ... = \mathbb{M}^{\Gamma_N},\\ Q_{1,l} = Q_{2,l} = ... = Q_{N,l}, \quad Q_{1,r} = Q_{2,r} = ... = Q_{N,r}. \end{aligned} \tag{3.8}$$

The conclusion follows directly from (3.5). ■

Proposition 3.2. *Let us consider the transmission condition S^m_{pade}. If the potential is zero, then the matrix $\mathcal{L}_h$ takes the form* (3.3) *and $\mathcal{L}_h$ is independent of time step n. In addition, if the subdomains are equal, then* (3.4) *is true.*

Proof. The proof is almost same as that of the Proposition 3.1. Using (2.20) and (2.7) gives

$$\begin{aligned} X^{j,1} &= -I + 2Q_{j,l}\Big(-i\sum_{s=0}^m a_s^m + i\sum_{s=1}^m a_s^m d_s^m \mathbb{D}_{j,n,s}^{-1}\mathbb{C}_j\Big)\\ &\qquad\times\Big(\mathbb{A}_{j,n} + i(\sum_{s=1}^m a_s^m)\cdot\mathbb{M}^{\Gamma_j} - \sum_{s=1}\mathbb{B}_{j,s}\mathbb{D}_{j,n,s}^{-1}\mathbb{C}_j\Big)^{-1}\mathbb{M}^{\Gamma_j}Q_{j,l}^\top,\\ X^{j,2} &= 2Q_{j,l}\Big(-i\sum_{s=0}^m a_s^m + i\sum_{s=1}^m a_s^m d_s^m \mathbb{D}_{j,n,s}^{-1}\mathbb{C}_j\Big)\Big(\mathbb{A}_{j,n} + i(\sum_{s=1}^m a_s^m)\cdot\mathbb{M}^{\Gamma_j} - \sum_{s=1}\mathbb{B}_{j,s}\mathbb{D}_{j,n,s}^{-1}\mathbb{C}_j\Big)^{-1}\mathbb{M}^{\Gamma_j}Q_{j,r}^\top,\\ X^{j,3} &= 2Q_{j,r}\Big(-i\sum_{s=0}^m a_s^m + i\sum_{s=1}^m a_s^m d_s^m \mathbb{D}_{j,n,s}^{-1}\mathbb{C}_j\Big)\Big(\mathbb{A}_{j,n} + i(\sum_{s=1}^m a_s^m)\cdot\mathbb{M}^{\Gamma_j} - \sum_{s=1}\mathbb{B}_{j,s}\mathbb{D}_{j,n,s}^{-1}\mathbb{C}_j\Big)^{-1}\mathbb{M}^{\Gamma_j}Q_{j,l}^\top, \end{aligned} \tag{3.9}$$

$$X^{j,4} = -I + 2Q_{j,r}\Big(-i\sum_{s=0}^{m} a_s^m + i\sum_{s=1}^{m} a_s^m d_s^m \mathbb{D}_{j,n,s}^{-1}\mathbb{C}_j\Big)$$
$$\times \Big(\mathbb{A}_{j,n} + i(\sum_{s=1}^{m} a_s^m)\cdot \mathbb{M}^{\Gamma_j} - \sum_{s=1} \mathbb{B}_{j,s}\mathbb{D}_{j,n,s}^{-1}\mathbb{C}_j\Big)^{-1}\mathbb{M}^{\Gamma_j}Q_{j,r}^{\top}.$$

Under these assumptions, we have (3.7) and (3.8). In addition, the matrix associated with the transmission condition S_{pade}^m satisfy

$$\mathbb{B}_{1,s} = \mathbb{B}_{2,s} = ... = \mathbb{B}_{N,s}, \quad \mathbb{C}_1 = \mathbb{C}_2 = ... = \mathbb{C}_N, \quad \mathbb{D}_{1,n,s} = \mathbb{D}_{2,n,s} = ... = \mathbb{D}_{N,n,s},\ s = 1,2,...m,$$

and $\mathbb{D}_{j,n,s} = Q_j(\frac{2i}{\Delta t}\mathbb{M}^{\Gamma_j} - \mathbb{S}^{\Gamma_j} + \mathbb{M}_{W_n}^{\Gamma_j} + d_s^m\mathbb{M}^{\Gamma_j})Q_j^{\top}$ is independent of time step n since $\mathbb{M}_{W_n}^{\Gamma_j} = 0$. ■

Intuitively, the Schrödinger operator without potential is a roughly approximating of the Schrödinger operator with potential:

$$i\partial_t + \partial_{xx} \approx i\partial_t + \partial_{xx} + \mathscr{V}.$$

In other words, $\mathscr{V}$ is a perturbation of the free Schrödinger operator. According to (3.2), if $\mathscr{V} = 0$, then we have

$$\mathcal{R}_{h,n}\cdot \mathbf{0} = \mathbf{d}_n,\ I - \mathcal{L}_h = I - (\mathcal{R}_{h,n} - \mathcal{R}_{h,n}\cdot\mathbf{0})$$

where $\mathbf{0}$ is the zero vector. Thus, the matrix $\mathcal{L}_h$ could be seen as an approximation of the nonlinear operator $\mathcal{R}_{h,n} - \mathcal{R}_{h,n}\cdot\mathbf{0}$ when $\mathscr{V}$ is not null:

$$P = I - \mathcal{L}_h \approx I - (\mathcal{R}_{h,n} - \mathcal{R}_{h,n}\cdot\mathbf{0}).$$

Based on the previous propositions, it is sufficient to compute only four blocks $X^{2,1}$, $X^{2,2}$, $X^{2,3}$ and $X^{2,4}$ to construct the preconditioner P. It will be shown in the following section that the construction can be implemented in a parallel way.

4. Parallel implementation

We present the parallel implementation of the classical algorithm (2.18) and the preconditioned algorithm (3.1) in this section. We fix one MPI process per subdomain [24]. We use the distributed matrix, vector and iterative linear system solver avalaible in PETSc library [6].

4.1. Classical algorithm

The discrete interface vector $\mathbf{g}_n^k$ is stored in a distributed manner in PETSc form. As shown by (4.1), $\mathbf{r}_{1,n}^k$ is located in MPI process 0, $\mathbf{l}_{j,n}^k$ and $\mathbf{r}_{j,n}^k$ are in MPI process $j-1$, $j = 2,3,...,N-1$ and $\mathbf{r}_{N,n}^k$ is in MPI process $N-1$.

$$\mathbf{g}_n^k = \begin{pmatrix} \mathbf{r}_{1,n}^k \\ \vdots \\ \mathbf{l}_{j,n}^k \\ \mathbf{r}_{j,n}^k \\ \vdots \\ \mathbf{r}_{N,n}^k \end{pmatrix} \begin{matrix} \}\text{MPI } 0 \\ \\ \Big\}\text{MPI } j-1 \\ \\ \}\text{MPI } N-1 \end{matrix} \tag{4.1}$$

As shown by (2.17) for $N = 3$, at iteration k, $\mathbf{v}_{j,n}^k, j = 1,2,...,N$ is computed on each subdomain locally and the boundary values are communicated.

4.2. **Preconditioned algorithm**

Thanks to the analysis yielded in previous section, we can build explicitly $\mathcal{L}_h$ with few computations. For the Robin transmission condition, it is based on the formulas (3.5). For the transmission condition S^m_{pade}, the idea is equivalent, but involves (3.9). According to the Proposition (3.1), the column s of $X^{2,1}$ and $X^{2,3}$ are

$$X^{2,1}\mathbf{e}_s = -\mathbf{e}_s - 2ip \cdot Q_{2,l}(\mathbb{A}_{j,n} + ip \cdot \mathbb{M}^{\Gamma_j})^{-1}\mathbb{M}^{\Gamma_2}Q_{2,l}^{\top}\mathbf{e}_s,$$
$$X^{2,3}\mathbf{e}_s = -2ip \cdot Q_{2,r}(\mathbb{A}_{j,n} + ip \cdot \mathbb{M}^{\Gamma_j})^{-1}\mathbb{M}^{\Gamma_2}Q_{2,l}^{\top}\mathbf{e}_s,$$

where $\mathbf{e}_s = (0, 0, ..., 1, ...0) \in \mathbb{C}^{N_T}$, all its elements are zero except the s-th, which is one. The element $\mathbb{M}^{\Gamma_2}Q_{2,l}^{\top}\mathbf{e}_s$ being a vector, it is necessary to compute one time the application of $(\mathbb{A}_{j,n} + ip \cdot \mathbb{M}^{\Gamma_j})^{-1}$ to vector. Similarly, we have

$$X^{2,2}\mathbf{e}_s = -2ip \cdot Q_{2,l}(\mathbb{A}_{j,n} + ip \cdot \mathbb{M}^{\Gamma_j})^{-1}\mathbb{M}^{\Gamma_2}Q_{2,r}^{\top}\mathbf{e}_s,$$
$$X^{2,4}\mathbf{e}_s = -\mathbf{e}_s - 2ip \cdot Q_{2,r}(\mathbb{A}_{j,n} + ip \cdot \mathbb{M}^{\Gamma_j})^{-1}\mathbb{M}^{\Gamma_2}Q_{2,r}^{\top}\mathbf{e}_s,$$

Let us recall that $\mathbb{A}_{j,n} = \frac{2i}{\Delta t}\mathbb{M}_j - \mathbb{S}_j$ for $V = 0$, $f = 0$. To know the first N_y columns of $X^{2,1}$, $X^{2,2}$, $X^{2,3}$ and $X^{2,4}$, we only have to compute $2N_y$ times the application of $(\mathbb{A}_{j,n} + ip \cdot \mathbb{M}^{\Gamma_j})^{-1}$ to vector. In other words, this amounts to solve the Schrödinger equation on a single subdomain $2N_y$ times to build the matrix $\mathcal{L}_h$. The resolutions are all independent. Therefore, we can solve them on different processors using MPI paradigm. We fix one MPI process per domain. To construct the matrix $\mathcal{L}_h$, we use the N MPI processes to solve the equation on a single subdomain (ex. $(0, T) \times \Omega_2$) $2N_y$ times. Each MPI process therefore solves the Schrödinger equation on a single subdomain maximum

$$N_{\text{mpi}} := [\frac{2N_y}{N}] + 1 \text{ times},$$

where $[x]$ is the integer part of x. This construction is therefore super-scalable in theory. Indeed, if N is doubled, then the size of subdomain is divided by two and N_{mpi} is also approximately halved.

Concerning the computational phase, the transpose of $\mathcal{L}_h$ is stored in a distributed manner using the library PETSc. As shown by (3.3), the first block column of $\mathcal{L}_h$ lies in MPI process 0. The second and third blocks columns are in MPI process 1, and so on for other processes. In addition, for any vector y, the vector $x := P^{-1}y$ is computed by solving the linear system

$$Px = y$$

with Krylov methods (GMRES or BiCGStab).

5. **Numerical results**

We implement the algorithms in a cluster consisting of 92 nodes (16 cores/node, Intel Sandy Bridge E5-2670, 32GB/node). We fix one MPI process per subdomain and 16 MPI processes per node. The communications are handled by Intel MPI. The linear systems related to (2.11), (2.2), (2.14) and (2.15), are solved with the LU direct method using MKL Pardiso library. The convergence condition for our algorithms is $\| \mathbf{g}_n^{k+1} - \mathbf{g}_n^k \| < 10^{-10}, n = 1, 2, ..., N_T$. The initial vectors are

- $\mathbf{g}_1^0 = \mathbf{0}$ or $\mathbf{g}_1^0 =$ random vector,
- $\mathbf{g}_n^0 = \lim\limits_{k \to \infty} \mathbf{g}_{n-1}^k, n = 2, 3, ..., N_T$.

Since the convergence properties for different time steps $n = 1, 2, ..., N_T$ are similar, we only consider the number of iterations required for convergence of the first time step $n = 1$. As mentioned in [17], using the zero initial vector could give wrong conclusions associated with the convergence. Thus, the zero vector is used when one wants to evaluate the computation time, while the random vector is used when comparing the transmission conditions. The theoretical optimal parameter p (resp. m) in the transmission condition Robin (resp. S^m_{pade}) is not at hand for us, we then seek the best parameter numerically for each case.

This section is composed of two subsections. The first one is devoted to the Schrödinger equation. In the second, we consider the simulation of Bose–Einstein condensates.

5.1. **Schrödinger equation**

We decompose the physical domain $(-16,16)\times(-8,8)$ into N equal subdomains without overlap. The final time T and the time step Δt are fixed to be $T=0.5$ and $\Delta t=0.01$ in this subsection. We consider two different meshes

$$\Delta x = 1/128,\ \Delta y = 1/8,$$
$$\Delta x = 1/2048,\ \Delta y = 1/64,$$

where the size of cell is $\Delta x\times\Delta y$. The potential and the initial datum (Figure 5.1) are

$$\mathscr{V} = |u|^2,\quad u_0(x,y) = e^{-x^2-y^2-0.5ix},$$

which give rise to a solution that propagates slowly to the negative side in y direction and undergoes dispersion. It is possible to solve numerically the Schrödinger equation on the entire domain Ω with the first mesh $(\Delta x = 1/128, \Delta y = 1/8)$ under the memory limitation (32G). We compare in this sub section the classical and the preconditioned algorithms, as well as the two transmission conditions.

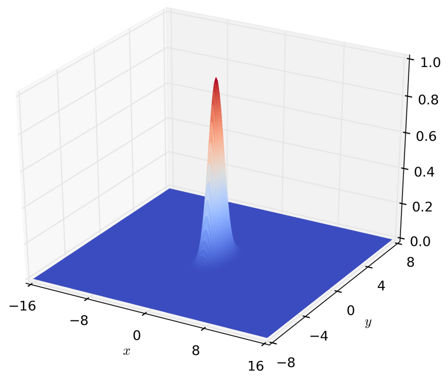


Figure 5.1. Initial datum $|u_0|$.

5.1.1. *Comparison of the classical algorithm and the preconditioned algorithm*

We are interested in observing the robustness, the number of iterations of the first time step, the computation time involving the Robin transmission condition and the transmission condition S^m_{pade}. The zero vector is used as the initial vector $\mathbf{g}^0_1$. We denote by N_{nopc} (resp. N_{pc}) the number of iterations required for convergence with the classical algorithm (resp. the preconditioned algorithm). T_{nopc} and T_{pc} denote the computation times of the classical algorithm and the preconditioned algorithm respectively. In addition, we denote by T^{ref} the computation time to solve numerically on a single processor the Schrödinger equation on the entire domain.

First, we consider a mesh with $\Delta x = 1/128$, $\Delta y = 1/8$. We make the tests for $N = 2, 4, 8, 16, 32$ subdomains. The convergence history of the first time step is presented in Figure 5.2 and Figure 5.3 for $N = 2$ (left) and $N = 32$ (right) where the two transmission conditions are considered. Table 5.1 and Table 5.2 show the number of iterations of the first time step and the computation times. We can see that all the algorithms are robust and scalable. The number of iterations is independent of number of subdomains. This independence has already been observed for one dimensional Schrödinger equation for not large N [21, 13]. In addition, the preconditioner allow to reduce number of iterations and computation time.

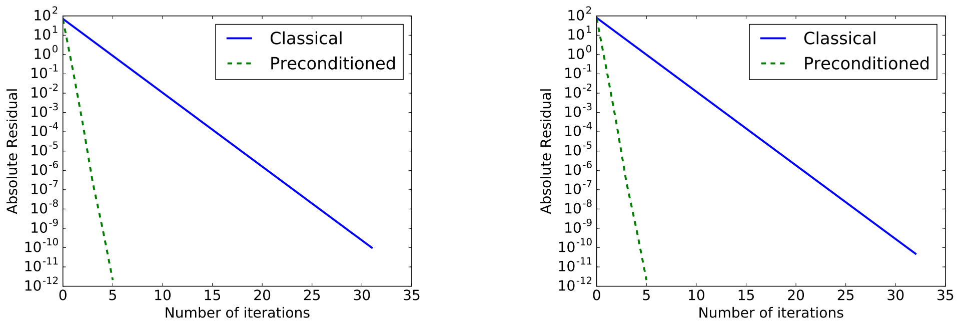


FIGURE 5.2. Convergence histories of the first time step for $N = 2$ (left) and $N = 32$ (right). The mesh is $\Delta x = 1/128$, $\Delta y = 1/8$ where the Robin transmission condition is considered.

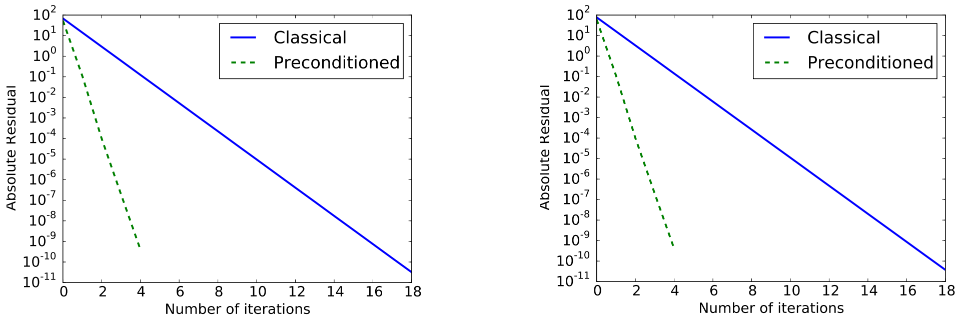


FIGURE 5.3. Convergence histories of the first time step for $N = 2$ (left) and $N = 32$ (right). The mesh is $\Delta x = 1/128$, $\Delta y = 1/8$ where the transmission condition S^m_{pade} is considered.

N	2	4	8	16	32
N_{nopc}	32	32	32	32	32
N_{pc}	6	6	6	6	6
T_{nopc}	1393.2	694.4	358.2	191.9	97.5
T_{pc}	252.2	123.9	62.9	33.8	17.5

TABLE 5.1. Number of iterations and total computation time (seconds) of the algorithms with the mesh $\Delta x = 1/128$, $\Delta y = 1/8$ where the Robin transmission condition is considered.

N	2	4	8	16	32
N_{nopc}	18	18	18	18	18
N_{pc}	5	5	5	5	5
T_{nopc}	1106.2	571.2	297.4	161.8	85.2
T_{pc}	356.2	180.5	92.1	50.3	26.6

Table 5.2. Number of iterations and total computation time (seconds) of the algorithms with the mesh $\Delta x = 1/128$, $\Delta y = 1/8$ where the transmission condition S^m_{pade} is considered.

Secondly, we reproduce the same tests with the mesh $\Delta x = 1/2048$, $\Delta y = 1/64$. The convergence history of the first time step, the total computation times are shown in Figure 5.5 and Table 5.3. The algorithms are both robust for $N = 1024$ except the classical algorithm with the Robin transmission condition, but not scalable from $N = 512$ to $N = 1024$. The classical algorithm loses scalability since if we use more subdomains to decompose Ω, then more iterations are required for convergence. Concerning the preconditioned algorithm, the computational time is larger with $N = 1024$ compared to $N = 512$ since the application of the preconditioner increases with larger N. However, the preconditioned algorithm is much more efficient since it can both reduce the number of iterations and the total computation times.

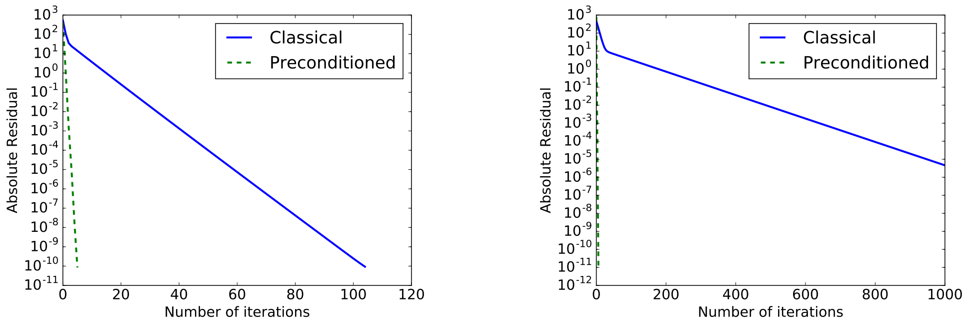


Figure 5.4. Convergence histories of the first time step for $N = 256$ (left) and $N = 1024$ (right) with the mesh $\Delta x = 1/2048$, $\Delta y = 1/64$ where the Robin transmission condition is considered.

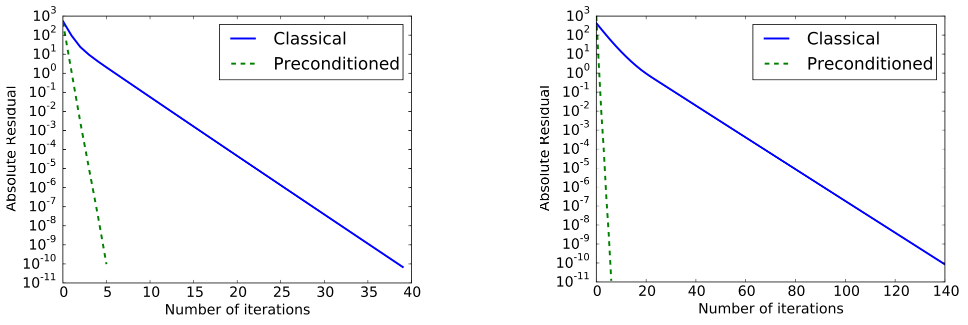


Figure 5.5. Convergence histories of the first time step for $N = 256$ (left) and $N = 1024$ (right) with the mesh $\Delta x = 1/2048$, $\Delta y = 1/64$ where the transmission condition S^m_{pade} is considered.

N		256	512	1024
Robin	Classical algorithm	7124.2	11702.7	-
	Preconditioned algorithm	492.4	347.9	350.2
S^m_{pade}	Classical algorithm	3582.4	2681.5	2516.6
	Preconditioned algorithm	596.9	376.1	441.9

$-$: The maximum number of iterations is fixed to 1000. The residual does not reach the convergence tolerance (see Figure 5.4).

Table 5.3. Computation times (seconds) of the algorithms with the mesh $\Delta x = 1/2048, \Delta y = 1/64$.

In addition, Table 5.1, Table 5.2 and Table 5.3 also show that in the framework of the classical algorithm, the transmission condition S^m_{pade} leads to better results than the Robin transmission condition in the context of computation time. Indeed, the classical algorithm with the transmission condition S^m_{pade} takes less iterations to converge. Instead, it is not true in the framework of the preconditioned algorithm. The numbers of iterations with the two transmission conditions are similar, but the transmission condition S^m_{pade} gives a more costly algorithm.

Finally, we show in Table 5.4 and Table 5.5 the number of iterations with different mesh $\Delta x, \Delta y$ and different time step size Δt with the Robin transmission condition and the transmission condition S^m_{pade} respectively. It can bee seen that the preconditioner is efficient to reduce the number of iterations.

		N_{nopc}	N_{pc}
$\Delta t = 10^{-2}$	$\Delta x = 1/64, \quad \Delta y = 1/4$	32	6
	$\Delta x = 1/128, \quad \Delta y = 1/8$	32	6
	$\Delta x = 1/256, \quad \Delta y = 1/16$	33	6
$\Delta t = 10^{-3}$	$\Delta x = 1/64, \quad \Delta y = 1/4$	32	5
	$\Delta x = 1/128, \quad \Delta y = 1/8$	34	5
	$\Delta x = 1/256, \quad \Delta y = 1/16$	34	5
$\Delta t = 10^{-4}$	$\Delta x = 1/64, \quad \Delta y = 1/4$	26	4
	$\Delta x = 1/128, \quad \Delta y = 1/8$	33	4
	$\Delta x = 1/256, \quad \Delta y = 1/16$	35	4

Table 5.4. Number of iterations of the first time step with different mesh $\Delta x, \Delta y$ and different time step size Δt where $N = 32$ and the Robin transmission condition is considered.

5.1.2. *Comparison of transmission conditions*

In this part, we compare numerically the transmission conditions Robin and S^m_{pade} in the framework of the two algorithms. The initial vector $\mathbf{g}^0_1$ here is a random vector to make sure that all the frequencies are included. The time step is fixed to be $\Delta t = 0.01$ and the mesh is $\Delta x = 1/128$, $\Delta y = 1/8$. Figure 5.6 and Figure 5.7 present the convergence histories of the first time step in the framework of the classical and the preconditioned algorithms with Robin and S^m_{pade} transmission conditions for $N = 2$ and $N = 32$ respectively. It can be seen that in the framework of the classical algorithm,

		N_{nopc}	N_{pc}
$\Delta t = 10^{-2}$	$\Delta x = 1/64, \quad \Delta y = 1/4$	18	6
	$\Delta x = 1/128, \quad \Delta y = 1/8$	18	5
	$\Delta x = 1/256, \quad \Delta y = 1/16$	19	6
$\Delta t = 10^{-3}$	$\Delta x = 1/64, \quad \Delta y = 1/4$	14	5
	$\Delta x = 1/128, \quad \Delta y = 1/8$	18	5
	$\Delta x = 1/256, \quad \Delta y = 1/16$	19	5
$\Delta t = 10^{-4}$	$\Delta x = 1/64, \quad \Delta y = 1/4$	14	4
	$\Delta x = 1/128, \quad \Delta y = 1/8$	17	4
	$\Delta x = 1/256, \quad \Delta y = 1/16$	14	4

Table 5.5. Number of iterations of the first time step with different mesh $\Delta x, \Delta y$ and different time step size Δt where $N = 32$ and the transmission condition S^m_{pade} is considered.

the transmission condition S^m_{pade} allows the algorithm to converge faster, while in the framework of preconditioned algorithm, they have similar histories of convergence. This observation indicates that the preconditioner P is a good approximation of the nonlinear operator $I - (\mathcal{R}_{h,n} - \mathcal{R}_{h,n} \cdot \mathbf{0})$. The influence of the transmission conditions is eliminated by the preconditioner. In addition, we could confirm the conclusion of the previous subsection: the preconditioner reduces a lot the number of iterations required for convergence.

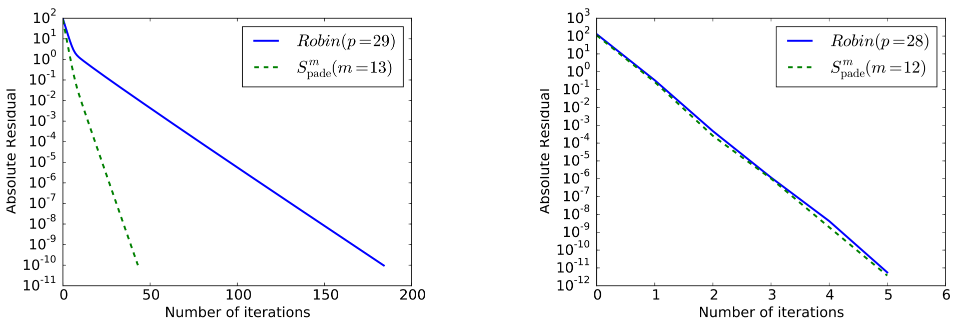


Figure 5.6. Convergence histories of the first time step of the classical algorithm (left) and the preconditioned algorithm for $N = 2$. The mesh is $\Delta x = 1/128$, $\Delta y = 1/8$.

5.1.3. *Influence of parameters*

In this subsection, we study the influence of parameters in the transmission conditions:

- the parameter m (order of Padé approximation) in the transmission condition S^m_{pade},
- the parameter p in the transmission condition Robin.

The time step and the mesh are fixed to be $\Delta t = 0.01$ and $\Delta x = 1/128$, $\Delta y = 1/8$.

Firstly, we consider the influence of m in the transmission condition S^m_{pade}. We present in Figure 5.8 and in Figure 5.9 the number of iterations in relation to the order of Padé approximation (m) in the

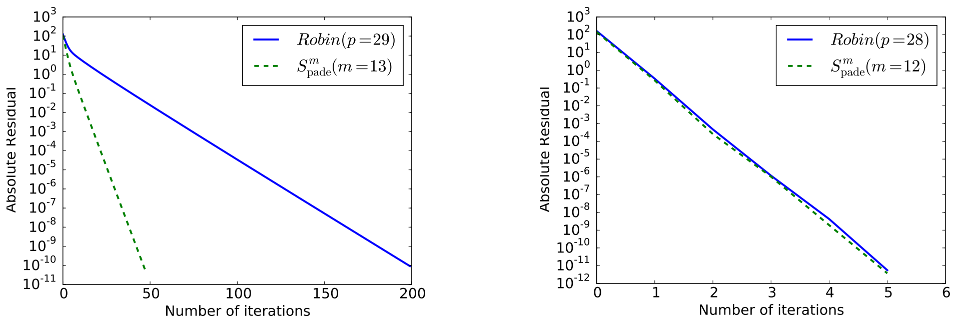


FIGURE 5.7. Convergence histories of the first time step of the classical algorithm (left) and the preconditioned algorithm for $N = 32$. The mesh is $\Delta x = 1/128$, $\Delta y = 1/8$.

framework of the classical and the preconditioned algorithms. Both of the zero vector and the random vector are considered as the initial vector in our tests.

- For the classical algorithm, if the initial vector is the zero vector, there exists an optimal parameter m. This observation is not consistent with our expectations since the higher order should make the algorithm converge faster. We believe that the zero initial vector gives us some inaccurate information.

- For the classical algorithm, if the initial vector is a random vector, the number of iterations first decreases then increases by increasing the order m. We however do not have yet an explanation for the relation between the convergence and the parameter m, which needs some more investigations.

- The parameter m is not very important for the preconditioned algorithm since the preconditioner hides the information about the order.

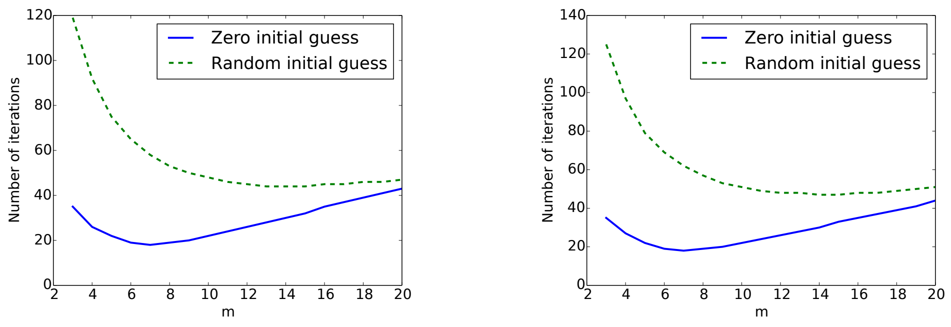


FIGURE 5.8. Number of iterations vs. parameter m for $N = 2$ (left) and $N = 32$ (right) in the framework of the classical algorithm.

Secondly, we study the influence of p in Robin transmission condition. The numbers of iterations are presented in Table 5.6 with different p for $N = 2$ and $N = 32$ (here only $p = 5, 10, ..., 50$ are shown). Both the classical algorithm and the preconditioned algorithm (Cls./Pd.), as well as the different initial vectors (zero or random) are considered. As can be seen, for the preconditioned algorithm, the number of iterations is almost the same in each case. For the classical algorithm, there exists an optimal p for each case.

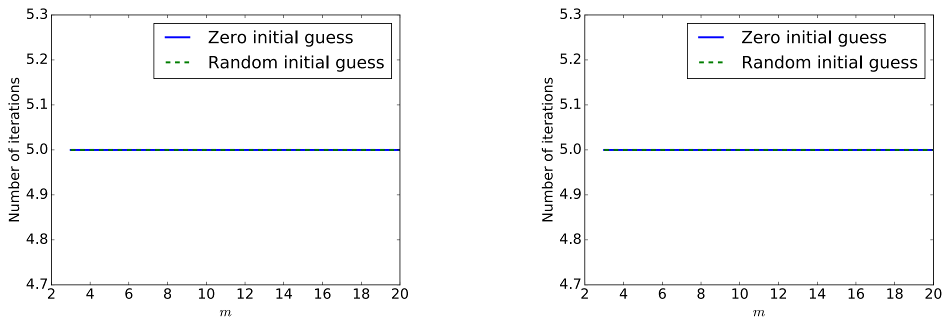


Figure 5.9. Number of iterations vs. parameter m for $N = 2$ (left) and $N = 32$ (right) in the framework of the preconditioned algorithm.

p	$N = 2$				$N = 32$			
	Zero		Random		Zero		Random	
	Cls.	Pd.	Cls.	Pd.	Cls.	Pd.	Cls.	Pd.
5	57	6	548	6	57	6	582	6
10	35	6	297	6	35	6	316	6
15	32	6	227	6	33	6	241	6
20	36	6	200	6	36	6	212	6
25	41	6	189	6	41	6	200	6
30	46	6	186	6	47	6	200	6
35	53	6	188	6	53	6	204	6
40	59	6	194	6	60	6	211	6
45	66	6	208	6	66	6	223	6
50	73	6	216	6	73	6	234	6

Table 5.6. Number of iterations vs. parameter p.

In conclusion, the use of the preconditioner allows to reduce both the number of iterations and the computation time. In addition, the preconditioned algorithm is not sensitive to the transmission conditions as well as the parameters in these transmission conditions.

5.2. **Simulation of Bose–Einstein condensates**

In this part, we apply the parallel algorithms to BEC simulation. Before comparing numerically the algorithms and making dynamic simulation of quantized vortex lattices, we recall some facts about BEC.

5.2.1. *Gross-Pitaevski equation*

A Bose–Einstein condensate (BEC) is a state of matter of a dilute gas of bosons cooled to temperatures very close to absolute zero. Under such conditions, a large fraction of bosons occupy the lowest quantum state, at which point macroscopic quantum phenomena become apparent. One of the models for BEC is the Gross–Pitaevskii (GPE) equation [7, 11, 9, 1]. In this paper, we consider the GPE equation defined on a bounded spatial domain with the same boundary conditions as (1.1):

$$\begin{cases} i\partial_t u + \frac{1}{2}\Delta u - V(x,y)u - \beta|u|^2 u + \omega \cdot L_z u = 0, \ (t,x,y) \in (0,T) \times \Omega, \\ u(0,x,y) = u_0(x,y). \end{cases} \tag{5.1}$$

The constant β describes the strength of the short-range two-body interactions (positive for repulsive interaction and negative for attractive interaction) in a condensate. The constant $\omega \in \mathbb{R}$ represents the angular velocity, the z-component of the angular momentum L_z is given by

$$L_z = -i(x\partial_y - y\partial_x).$$

The potential here is

$$V(x,y) = \frac{1}{2}(\gamma_x^2 x^2 + \gamma_y^2 y^2), \ \gamma_x, \gamma_y \in \mathbb{R}.$$

The GPE equation is a type of nonlinear Schrödinger equation. One of the difficulties in the simulation of Bose–Einstein condensates derives from the term of rotation. Recently, the authors of [10] introduced a coordinate transformation that allows to write the GPE equation in this new coordinates as a nonlinear Schrödinger equation (5.3) with a time-dependent potential but without the rotation term. Thus, the algorithms that we presented in the previous sections are applicable for GPE equation. For $\forall t \geqslant 0$, the orthogonal rotational matrix $A(t)$ is defined by

$$A(t) = \begin{pmatrix} \cos(\omega t) & \sin(\omega t) \\ -\sin(\omega t) & \cos(\omega t) \end{pmatrix}.$$

The transformed Lagrange coordinate $(\widetilde{x}, \widetilde{y})$ is then defined as

$$\begin{pmatrix} \widetilde{x} \\ \widetilde{y} \end{pmatrix} = A^{-1}(t) \begin{pmatrix} x \\ y \end{pmatrix} = A^{\top}(t) \begin{pmatrix} x \\ y \end{pmatrix}. \tag{5.2}$$

In this new coordinate, the GPE equation (5.1) could be written as

$$\begin{cases} i\partial_t \widetilde{u} + \frac{1}{2}\Delta \widetilde{u} - V_t(t,\widetilde{x},\widetilde{y})\widetilde{u} - \beta|\widetilde{u}|^2 \widetilde{u} = 0, \ t \in (0,T), \\ \widetilde{u}(0,\widetilde{x},\widetilde{y}) = \widetilde{u}_0(\widetilde{x},\widetilde{y}), \end{cases} \tag{5.3}$$

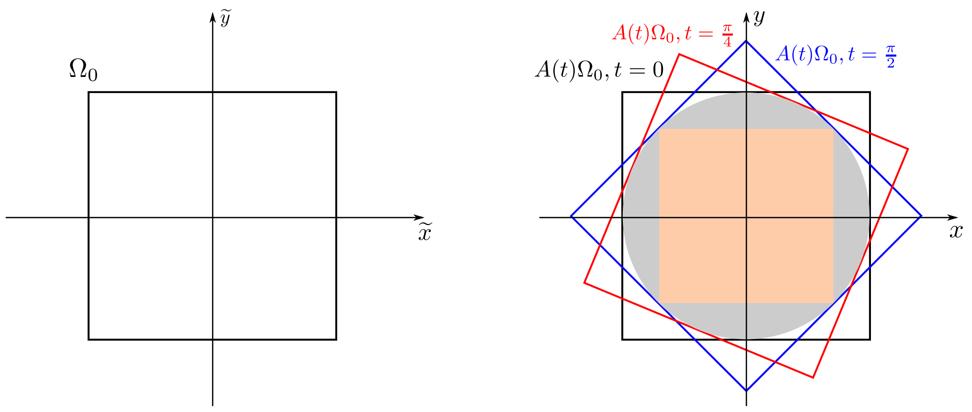
where

$$\widetilde{u}(t,\widetilde{x},\widetilde{y}) := u(t,x,y), \ V_t(t,\widetilde{x},\widetilde{y}) := V(x,y), \text{ where } (x,y)^{\top} = A(t)(\widetilde{x},\widetilde{y})^{\top}. \tag{5.4}$$

Formally, the only difference between the equation (5.3) and the Schrödinger equation (1.1) is the constant in front of the Laplace operator Δ. Thus, we could directly apply the domain decomposition algorithms to the equation (5.3) on the spatial domain $\Omega = (x_l, x_r) \times (y_b, y_u)$. The Robin transmission condition and the transmission condition S^m_{pade} are given by (2.3), (2.4) and (2.5). A minor modification concerns the constant before the operator Δ_{Γ_j} in (2.5) which is $\frac{1}{2}$ here.

Once the solution $\widetilde{u}$ is computed numerically, it is possible to reconstruct the solution u by (5.4). At time t, the computational domain of $\widetilde{u}(t,\widetilde{x},\widetilde{y})$ is $\Omega = (x_l, x_r) \times (y_b, y_u)$ and the computational domain of $u(t,x,y)$ is $A(t)\Omega$ (see figure 5.10). The domains $A(t)\Omega$ for $t \geqslant 0$ share a common disc. The values of $u(t,x,y)$ within the maximum square (the valid zone) are all in the disc, which could be computed by interpolation. The valid zone is

$$(\frac{x_l}{\sqrt{2}}, \frac{x_r}{\sqrt{2}}) \times (\frac{y_b}{\sqrt{2}}, \frac{y_u}{\sqrt{2}}).$$



(A) $(\widetilde{x}, \widetilde{y}) \in \Omega$.

(B) $(x, y) \in A(t)\Omega$.

Figure 5.10. (a) The computational domain Ω. (b) The domain $A(t)\Omega$ at some different times: $t = 0$, $t = \pi/4$ and $t = \pi/2$ where $\omega = 0.5$.

5.2.2. *Comparison of algorithms*

In this part, we fix the physical domain to be $\Omega = (-16, 16) \times (-16, 16)$. The initial datum is taken as a Gaussian

$$u_0(x, y) = \frac{1}{\pi^{1/4}} e^{\frac{-(x^2+2y^2)}{2}}, \ (x, y) \in \mathbb{R}^2,$$

where the coefficients are $\omega = 0.4$ and $\beta = 10.15$. The time step is fixed as $\Delta t = 0.0001$. Firstly, we use a wide mesh $\Delta x = \Delta y = 1/32$, which generates 1024×1024 unknowns on Ω. It is possible to solve the GPE equation (5.3) on the complete domain Ω under our memory limitation (32G) without using the parallel algorithms (classical or preconditioned algorithm). However, the computation time could be very long. Thus, we use here a small final time $T = 0.1$. Using the same notations as in the previous sections, we show in Table 5.7 the computation times of the two algorithms with Robin and S^m_{pade} transmission conditions. Since the boundary condition imposed on Ω is associated with the transmission operator, the reference times T^{ref} for the two transmission condition are different. In BEC simulation, a small time step is necessary. According to our experiments, when a small Δt is considered, a large m in S^m_{pade} transmission condition is needed to ensure fast convergence. Thus, the use of the transmission condition S^m_{pade} is much more expensive than the Robin transmission condition. We can also see that the computation times of the classical algorithm (T_{pc}) and the preconditioned algorithm (T_{nopc}) are scalable.

N		2	4	8	16	32
Robin, $p = 180$	T^{ref}	5.68				
	T_{nopc}	5.68	2.66	1.28	0.68	0.33
	T_{pc}	3.49	1.60	0.77	0.44	0.24
S^m_{pade}, $m = 76$	T^{ref}	8.41				
	T_{nopc}	> 20	10.70	7.40	5.07	4.23
	T_{pc}	6.30	3.52	2.30	1.68	1.37

Table 5.7. Computation time in hours with the mesh $\Delta x = \Delta y = 1/32$.

We make the tests with a finer mesh $\Delta x = 1/1024$, $\Delta y = 1/64$ with the Robin transmission condition since it has been seen that the implementation with the transmission condition S^m_{pade} is much more expensive than with the Robin transmission condition in the context of Gross-Pitaevski equation. The complete domain is decomposed into $N = 128, 256, 512, 1024$ subdomains. The computation times are presented in Table 5.8. We could see that the both algorithms are scalable. In addition, the preconditioner allows to reduce the total computation time. However, since the implementation of the preconditioner consumes memory, the memory is not sufficient in the case $N = 128$.

N	128	256	512	1024
T_{nopc}, $p = 95$	19.3	8.8	5.0	2.1
T_{pc}, $p = 95$	*	2.3	1.6	0.8

*: the memory is not sufficient.

Table 5.8. Computation time in hours with the mesh $\Delta x = 1/1024$, $\Delta y = 1/64$.

5.2.3. *Dynamic simulation of quantized vortex lattices*

According to the studies in the previous subsection, we apply the algorithms with the Robin transmission condition to study the dynamics of quantized vortex lattices for BEC with rotation. In this simulation, the nonlinear potential and the parameters are

$$V(x, y) = \frac{1}{2}(x^2 + y^2),\ \beta = 1000,\ \omega = 0.9.$$

The initial solution u_0 is a stationary vortex lattice [7, 4]. The stationary solution ϕ of (5.1) is defined as

$$u(t, x, y) = \phi(x, y)e^{-i\mu t}, \tag{5.5}$$

where μ is the chemical condensation potential. By substituing (5.5) in (5.1), we have

$$\mu\phi = -\frac{1}{2}\phi + V\phi + \beta|\phi|^2\phi - \omega L_z\phi,$$

with the constraint of normalisation

$$||\phi||_2^2 = \int_{\mathbb{R}^2} |\phi(x, y)|^2 dxdy = 1.$$

This is therefore a nonlinear eigenvalue problem. The eigenvalue μ can be computed from its corresponding eigenvector ϕ by

$$\mu_{\beta,\omega}(\phi) = E_{\beta,\omega}(\phi) + \frac{\beta}{4}\int_{\mathbb{R}^2} |\phi(x, y)|^4 dxdy,$$

where

$$E_{\beta,\omega}(\phi) = \frac{1}{2}\int_{\mathbb{R}^2}(|\nabla\phi|^2 + V|\phi|^2 + \beta|\phi|^4 - \omega\overline{\phi}L_z\phi)dxdy. \tag{5.6}$$

The ground state of a BEC is defined as the solution of minimization problem, denoted by ϕ_g,

$$E_{\beta,\omega}(\phi_g) = \min_{\phi\in S} E_{\beta,\omega}(\phi),$$

where $S = \{\phi|\ ||\phi||_2 = 1, E_{\beta,\omega} < \infty\}$.

For our simulation, we take the solution of minimization problem as the datum initial

$$u_0(x, y) = \phi_g(x, y). \tag{5.7}$$

It is computed by BESP method (Backward Euler Sine Pseudospectral) [8] using GPELab [4], a matlab toolbox developed for the computation of the ground states and the dynamics of quantum systems modeled by GPE equations.

The complete domain $\Omega = (-16, 16) \times (-16, 16)$ is decomposed into $N = 32$ subdomains. We fix the time step as $\Delta t = 0.0001$. The mesh is $\Delta x = \Delta y = 1/32$. The parameter p here is $p = 180$. Figure 5.11 shows the contours of the solution $|u(t, x, y)|^2$ at some different times. The solution is illustrated in the valid zone $(-16/\sqrt{2}, 16/\sqrt{2}) \times (-16/\sqrt{2}, (16 - \Delta y)/\sqrt{2})$. The total computation time is about 16 hours.

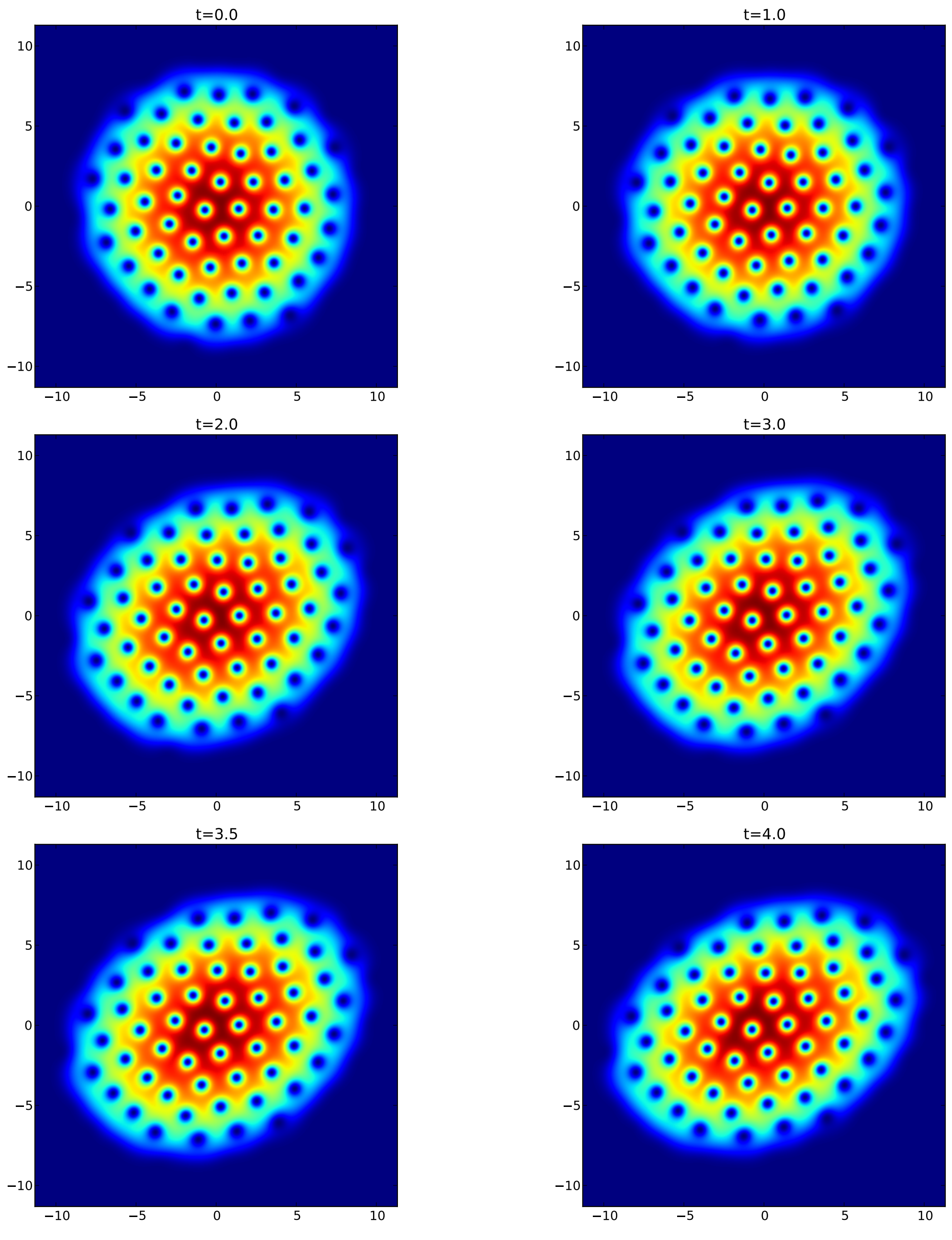


Figure 5.11. Contours of solution $|u(t, x, y)|^2$ at some different times.

6. Conclusion and perspective

We applied the optimized Schwarz method to the two dimensional nonlinear Schrödinger equation and GPE equation. We proposed a preconditioned algorithm which allows to reduce the number of iterations and the computation time. According to the numerical tests, the preconditioned algorithm is not sensitive to the transmission conditions (Robin, S^m_{pade}) and the parameters in these conditions. In addition, the parallel algorithms are applied to the BEC simulation. We can obtain an accurate solution by using the parallel algorithms and the computation time of the preconditioned algorithm is less than the classical one.

One perspective could be to use a partially constructed $I - \mathcal{L}_h$ as the preconditioner in the context of the multilevel preconditioner. The construction and the implementation should be less expensive.

Acknowledgements

We acknowledge Pierre Kestener (Maison de la Simulation Saclay France) for the discussions about the parallel programming. This work was partially supported by the French ANR grant ANR-12-MONU-0007-02 BECASIM (Modèles Numériques call). The first author also acknowledges support from the French ANR grant BonD ANR-13-BS01-0009-01.

References

[1] X. Antoine, W. Bao, and C. Besse. Computational methods for the dynamics of the nonlinear Schrödinger/Gross–Pitaevskii equations. *Comput. Phys. Commun.*, 184(12):2621–2633, 2013.

[2] X. Antoine, C. Besse, and P. Klein. Absorbing Boundary Conditions for the Two-Dimensional Schrödinger Equation With an Exterior Potential Part I: Construction and a Priori Estimates. *Math. Model. Methods Appl. Sci.*, 22(10), 2012.

[3] X. Antoine, C. Besse, and P. Klein. Absorbing boundary conditions for the two-dimensional Schrödinger equation with an exterior potential. Part II: Discretization and numerical results. *Numer. Math.*, 125(2):191–223, 2013.

[4] X. Antoine and R. Duboscq. Computer Physics cations GPELab , a Matlab Toolbox to solve Gross–Pitaevskii Equations I : computation of stationary solutions. *Comput. Phys. Commun.*, 185(11):2969–2991, 2014.

[5] X. Antoine, E. Lorin, and A. D. Bandrauk. Domain Decomposition Method and High-Order Absorbing Boundary Conditions for the Numerical Simulation of the Time Dependent Schrödinger Equation with Ionization and Recombination by Intense Electric Field. *J. Sci. Comput.*, pages 1–27, 2014.

[6] S. Balay, M. F. Adams, J. Brown, P. Brune, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, K. Rupp, B. F. Smith, and H. Zhang. PETSc Users Manual. Technical Report ANL-95/11 - Revision 3.4, Argonne National Laboratory, 2013.

[7] W. Bao and Y. Cai. Mathematical theory and numerical methods for Bose-Einstein condensation. *Kinet. Relat. Model.*, 6(1):1–135, December 2012.

[8] W. Bao, I.-L. Chern, and F. Y. Lim. Efficient and spectrally accurate numerical methods for computing ground and first excited states in Bose–Einstein condensates. *J. Comput. Phys.*, 219(2):836–854, 2006.

[9] W. Bao and Q. Du. Computing the ground state solution of Bose–Einstein condensates by a normalized gradient flow. *SIAM J. Sci. Comput.*, 25(5):1674–1697, 2004.

[10] W. Bao, D. Marahrens, Q. Tang, and Y. Zhang. A Simple and Efficient Numerical Method for Computing the Dynamics of Rotating Bose–Einstein Condensates via Rotating Lagrangian Coordinates. *SIAM J. Sci. Comput.*, 35(6), 2013.

[11] W. Bao, P. A. Markowich, and H. Wang. Ground, Symmetric and Central Vortex States in Rotating Bose-Einstein Condensates. *Commun. Math. Sci.*, 3(1):57–88, 2005.

[12] C. Besse and F. Xing. Domain decomposition algorithms for two dimensional linear Schrödinger equation. `https://arxiv.org/abs/1506.05639`, 2015.

[13] C. Besse and F. Xing. Schwarz waveform relaxation method for one dimensional Schrödinger equation with general potential. *Numerical Algorithms, accepted*, 2016.

[14] Y. Boubendir, X. Antoine, and C. Geuzaine. A quasi-optimal non-overlapping domain decomposition algorithm for the Helmholtz equation. *J. Comput. Phys.*, 231(2):262–280, 2012.

[15] A. Durán and J.-M. Sanz-Serna. The numerical integration of relative equilibrium solutions. The nonlinear Schrodinger equation. *IMA J. Numer. Anal.*, 20(2):235–261, April 2000.

[16] M. J. Gander. Optimized Schwarz Methods. *SIAM J. Numer. Anal.*, 44(2):699–731, January 2006.

[17] M. J. Gander. Schwarz methods over the course of time. *Electron. Trans. Numer. Anal.*, 31:228–255, 2008.

[18] M. J. Gander and L. Halpern. Optimized Schwarz Waveform Relaxation Methods for Advection Reaction Diffusion Problems. *SIAM J. Numer. Anal.*, 45(2):666–697, January 2007.

[19] M. J. Gander and L. Halpern. *Méthodes de décomposition de domaine.* Encyclopédie électronique pour les ingénieurs, 2012.

[20] M. J. Gander, F. Magoules, and F. Nataf. Optimized Schwarz methods without overlap for the Helmholtz equation. *SIAM J. Sci. Comput.*, 24(1):38–60, 2002.

[21] L. Halpern and J. Szeftel. Optimized and quasi-optimal Schwarz waveform relaxation for the one dimensional Schrödinger equation. *Math. Model. Methods Appl. Sci.*, 20(12):2167–2199, 2010.

[22] P.-L. Lions. On the Schwarz alternating method. III: a variant for nonoverlapping subdomains. *Third Int. Symp. domain Decompos. methods Partial Differ. equations*, 6:202–223, 1990.

[23] S. Loisel. Condition Number Estimates for the Nonoverlapping Optimized Schwarz Method and the 2-Lagrange Multiplier Method for General Domains and Cross Points. *SIAM J. Numer. Anal.*, 51(6):3062–3083, 2013.

[24] Message Passing Interface Forum. MPI : A Message-Passing Interface Standard Version 3.0, 2012. `https://spcl.inf.ethz.ch/Publications/index.php?pub=160`.

[25] F. Nataf and F. Rogier. Factorization of the convection-diffusion operator and a (possibly) non overlapping Schwarz method. *Contemp. Math.*, 1994.