Journal of Computational Physics 343 (2017) 92–109

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Efficient spectral computation of the stationary states of rotating Bose–Einstein condensates by preconditioned nonlinear conjugate gradient methods

Xavier Antoine[a], Antoine Levitt[b,c], Qinglin Tang[a,d,*]

[a] Institut Elie Cartan de Lorraine, Université de Lorraine, UMR 7502, Inria Nancy-Grand Est, SPHINX Team, F-54506 Vandoeuvre-lès-Nancy Cedex, France
[b] Inria Paris, F-75589 Paris Cedex 12, France
[c] Université Paris-Est, CERMICS (ENPC), F-77455 Marne-la-Vallée, France
[d] Beijing Computational Science Research Center, No 10 East Xibeiwang Road, Beijing 100193, PR China

A R T I C L E   I N F O

Article history:
Received 6 November 2016
Received in revised form 20 March 2017
Accepted 15 April 2017
Available online 21 April 2017

Keywords:
Bose–Einstein condensation
Rotating Gross–Pitaevskii equation
Stationary states
Fourier spectral method
Steepest descent
Conjugate gradient
Optimization algorithms on Riemannian manifolds
Preconditioner

A B S T R A C T

We propose a preconditioned nonlinear conjugate gradient method coupled with a spectral spatial discretization scheme for computing the ground states (GS) of rotating Bose–Einstein condensates (BEC), modeled by the Gross–Pitaevskii Equation (GPE). We first start by reviewing the classical gradient flow (also known as *imaginary time* (IMT)) method which considers the problem from the PDE standpoint, leading to numerically solve a dissipative equation. Based on this IMT equation, we analyze the forward Euler (FE), Crank–Nicolson (CN) and the classical backward Euler (BE) schemes for linear problems and recognize classical power iterations, allowing us to derive convergence rates. By considering the alternative point of view of minimization problems, we propose the preconditioned steepest descent (PSD) and conjugate gradient (PCG) methods for the GS computation of the GPE. We investigate the choice of the preconditioner, which plays a key role in the acceleration of the convergence process. The performance of the new algorithms is tested in 1D, 2D and 3D. We conclude that the PCG method outperforms all the previous methods, most particularly for 2D and 3D fast rotating BECs, while being simple to implement.

© 2017 Elsevier Inc. All rights reserved.

1. Introduction

Bose–Einstein Condensates (BECs) were first predicted theoretically by S.N. Bose and A. Einstein, before being realized experimentally in 1995 [4,20,28,31]. This state of matter has the interesting feature that macroscopic quantum physics properties can emerge and be observed in laboratory experiments. The literature on BECs has grown extremely fast over the last 20 years in atomic, molecular, optics and condensed matter physics, and applications from this new physics are starting to appear in quantum computation for instance [22]. Among the most important directions, a particular attention has been paid towards the understanding of the nucleation of vortices [1,21,37–39,41,48] and the properties of dipolar gases [13,14] or multi-components BECs [11–13]. At temperatures T which are much smaller than the critical temperature T_c, the macroscopic behavior of a BEC can be well described by a condensate wave function ψ which is solution to a Gross–Pitaevskii

* Corresponding author.
E-mail addresses: xavier.antoine@univ-lorraine.fr (X. Antoine), antoine.levitt@inria.fr (A. Levitt), tqltql2010@gmail.com (Q. Tang).
URL: http://iecl.univ-lorraine.fr/~xantoine/ (X. Antoine).

http://dx.doi.org/10.1016/j.jcp.2017.04.040
0021-9991/© 2017 Elsevier Inc. All rights reserved.

Equation (GPE). Being able to compute efficiently the numerical solution of such a class of equations is therefore extremely useful. Among the most crucial questions are the calculations of stationary states, i.e. ground/excited states, and of the real-time dynamics [5,9,13,34,35].

To fully analyze a representative and nontrivial example that can be extended to other more general cases, we consider in this paper a BEC that can be modeled by the rotating (dimensionless) GPE. In this setting, the computation of a ground state of a d-dimensional BEC takes the form of a constrained minimization problem:

$$\text{Find} \qquad \phi \in L^2(\mathbb{R}^d) \qquad \text{s.t.} \qquad \phi \in \underset{\|\phi\|=1}{\arg\min}\, E(\phi), \tag{1.1}$$

here $\|\phi\| = \int_{\mathbb{R}^d} |\phi|^2$ is the standard L^2-norm and E is the associated energy functional. Several approaches can be developed for computing the stationary state solution to the rotating GPE. For example, some techniques are based on appropriate discretizations of the continuous normalized gradient flow/imaginary-time formulation [3,7,9,13,15,19,25,26,49], leading to various iterative algorithms. These approaches are general and can be applied to many situations (dipolar interactions, multi-components GPEs...). We refer for instance to the recent freely distributed Matlab solver GPELab that provides the stationary states computation [6] (and real-time dynamics [8]) for a wide variety of GPEs based on the so-called BESP (Backward Euler pseudo-Spectral) scheme [7,9,13,15] (see also Sections 4 and 6). Other methods are related to the numerical solution of the nonlinear eigenvalue problem [32,46] or on optimization techniques under constraints [17,23,29,30]. As we will see below in Section 4, some connections exist between these approaches. Finally, a regularized Newton-type method was proposed recently in [47].

Optimization problems with orthogonal or normalization constraints also occur in different branches of computational science. An elementary but fundamental example is the case of a quadratic energy, where solving the minimization problem is equivalent to finding an eigenvector associated with the lowest eigenvalue of the symmetric matrix representing the quadratic form. A natural generalization is a class of orthogonalized minimization problems, which for a quadratic energy reduce to finding the N first eigenvectors of a matrix. Many problems in electronic structure theory are of this form, including the celebrated Kohn–Sham and Hartree–Fock models [24,44]. Correspondingly, a large amount of effort has been devoted to finding efficient discretization and minimization schemes. A workhorse of these approaches is the nonlinear preconditioned conjugate gradient method, developed in the 80s [40], as well as several variants of this (the Davidson algorithm, or the LOBPCG method [36]).

Although similar, there are significant differences between the mathematical structure of the problem in electronic structure theory and the Gross–Pitaevskii equation. In some respects, solving the GPE is easier: there is only one wavefunction (or only a few for multi-species gases), and the nonlinearity is often local (at least when dipolar effects are not taken into account), with a simpler mathematical form than many electronic structure models. On the other hand, the Gross–Pitaevskii equation describes the formation of vortices: the discretization schemes must represent these very accurately, and the energy landscape presents shallower minima, leading to a more difficult optimization problem.

In the present paper, we consider the constrained nonlinear conjugate gradient method for solving the rotating GPE (Section 2) with a pseudo-Spectral discretization scheme (see Section 3). This approach provides an efficient and robust way to solve the minimization problem. Before introducing the algorithm, we review in Section 4 the discretization of the gradient flow/imaginary-time equation by standard schemes (explicit/implicit Euler and Crank–Nicolson schemes). This enables us to make some interesting and meaningful connections between these approaches and some techniques related to eigenvalue problems, such as the power method. In Sections 5.1 and 5.2, we introduce the projected preconditioned steepest descent (PSD) and preconditioned conjugate gradient (PCG) methods for solving the minimization problem on the Riemannian manifold defined by the spherical constraints. In particular, we provide some formulae to compute the stepsize arising in such iterative methods to get the energy decay assumption fulfilled. The stopping criteria and convergence analysis are discussed in Sections 5.3 and 5.4. We then investigate the design of preconditioners (Section 5.5). In particular, we propose a new simple symmetrical combined preconditioner, denoted by P_C. In Section 6, we consider the numerical study of the minimization algorithms for the 1D, 2D and 3D GPEs (without and with rotation). We first propose in Section 6.1 a thorough analysis in the one-dimensional case. This shows that the PCG approach with combined preconditioner P_C and pseudo-Spectral discretization, called PCG$_C$ method, outperforms all the other approaches, most particularly for very large nonlinearities. In Sections 6.2 and 6.3, we confirm these properties in 2D and 3D, respectively, and show how the PCG$_C$ algorithm behaves with respect to increasing the rotation speed. Finally, Section 7 provides a conclusion.

2. Definitions and notations

For the considered minimization problem (1.1), the energy functional E is defined by

$$E(\phi) = \int_{\mathbb{R}^d} \left[\frac{1}{2}|\nabla\phi|^2 + V(\mathbf{x})|\phi|^2 + \frac{\eta}{2}|\phi|^4 - \omega\phi^* L_z\phi \right],$$

where V is an external potential, η is the nonlinearity strength, ω is the rotation speed, and $L_z = i(y\partial_x - x\partial_y)$ is the angular momentum operator.

A direct computation of the gradient of the energy leads to

$$\nabla E(\phi) = 2H_\phi \phi, \quad \text{with} \quad H_\phi = -\frac{1}{2}\Delta + V + \eta|\phi|^2 - \omega L_z$$

the mean-field Hamiltonian. We can compute the second-order derivative as

$$\frac{1}{2}\nabla^2 E(\phi)[f,f] = \left\langle f, H_\phi f\right\rangle + \eta \operatorname{Re}\left\langle \phi^2, f^2\right\rangle.$$

We introduce $\mathcal{S} = \{\phi \in L^2(\mathbb{R}^d), \|\phi\| = 1\}$ as the spherical manifold associated to the normalization constraint. Its tangent space at a point $\phi \in \mathcal{S}$ is $T_\phi\mathcal{S} = \{h \in L^2(\mathbb{R}^d), \operatorname{Re}\langle\phi, h\rangle = 0\}$, and the orthogonal projection M_ϕ onto this space is given by $M_\phi h = h - \operatorname{Re}\langle\phi, h\rangle\,\phi$.

The Euler–Lagrange equation (first-order necessary condition) associated with our problem states that, at a minimum $\phi \in \mathcal{S}$, the projection of the gradient on the tangent space is zero, which is equivalent to

$$H_\phi \phi = \lambda\phi,$$

where $\lambda = \langle H_\phi\phi, \phi\rangle$ is the Lagrange multiplier associated to the spherical constraint, and is also known as the chemical potential. Therefore, the minimization problem can be seen as a nonlinear eigenvalue problem. The second-order necessary condition states that, for all $h \in T_\phi\mathcal{S}$,

$$\frac{1}{2}\nabla^2 E(\phi)[h,h] - \lambda\,\|h\|^2 \geq 0.$$

For a linear problem ($\eta = 0$) and for problems where the nonlinearity has a special structure (for instance, the Hartree–Fock model), this implies that λ is the lowest eigenvalue of H_ϕ (a property known as the *Aufbau* principle in electronic structure theory). This property is not satisfied here.

3. Discretization

To find a numerical solution of the minimization problem, the function $\phi \in L^2(\mathbb{R}^d)$ must be discretized. The presence of vortices in the solution imposes strong constraints on the discretization, which must be accurate enough to resolve fine details. Several discretization schemes have been used to compute the solution to the GPE, including high-order finite difference schemes or finite element schemes with adaptive meshing strategies [29,30]. Here, we consider a standard pseudo-spectral discretization based on Fast Fourier Transforms (FFTs) [7,9,15,49].

We truncate the wave function ϕ to a square domain $[-L, L]^d$, with periodic boundary conditions, and discretize ϕ with the same even number of grid points M in any dimension. These two conditions can of course be relaxed to a different domain size L and number of points M in each dimension, at the price of more complex notations. We describe our scheme in 2D, its extension to other dimensions being straightforward. We introduce a uniformly sampled grid: $\mathcal{D}_M := \{\mathbf{x}_{k_1,k_2} = (x_{k_1}, y_{k_2})\}_{(k_1,k_2)\in\mathcal{O}_M}$, with $\mathcal{O}_M := \{0, \dots, M-1\}^2$, $x_{k_1+1} - x_{k_1} = y_{k_2+1} - y_{k_2} = h$, with mesh size $h = 2L/M$, M an even number. We define the discrete Fourier frequencies (ξ_p, μ_q), with $\xi_p = p\pi/L, -M/2 \leq p \leq M/2 - 1$, and $\mu_q = q\pi/L, -M/2 \leq q \leq M/2 - 1$. The pseudo-spectral approximations $\widetilde{\phi}$ of the function ϕ in the x- and y-directions are such that

$$\widetilde{\phi}(t,x,y) = \frac{1}{M}\sum_{p=-M/2}^{M/2-1} \widehat{\widetilde{\phi}}_p(t,y)e^{i\xi_p(x+L)}, \qquad \widetilde{\phi}(t,x,y) = \frac{1}{M}\sum_{q=-M/2}^{M/2-1} \widehat{\widetilde{\phi}}_q(t,x)e^{i\mu_q(y+L)},$$

where $\widehat{\widetilde{\phi}}_p$ and $\widehat{\widetilde{\phi}}_q$ are respectively the Fourier coefficients in the x- and y-directions

$$\widehat{\widetilde{\phi}}_p(t,y) = \sum_{k_1=0}^{M-1} \widetilde{\phi}_{k_1}(t,y)e^{-i\xi_p(x_{k_1}+L)}, \qquad \widehat{\widetilde{\phi}}_q(t,x) = \sum_{k_2=0}^{M-1} \widetilde{\phi}_{k_2}(t,x)e^{-i\mu_q(y_{k_2}+L)}.$$

The following notations are used: $\widetilde{\phi}_{k_1}(t,y) = \widetilde{\phi}(t, x_{k_1}, y)$ and $\widetilde{\phi}_{k_2}(t,x) = \widetilde{\phi}(t,x,y_{k_2})$. In order to evaluate the operators, we introduce the matrices

$$\mathbb{I}_{k_1,k_2} := \delta_{k_1,k_2}, \qquad [[V]]_{k_1,k_2} := V(\mathbf{x}_{k_1,k_2}), \qquad [[|\phi|^2]]_{k_1,k_2} = |\phi_{k_1,k_2}|^2,$$

for $(k_1,k_2) \in \mathcal{O}_M$, and δ_{k_1,k_2} being the Dirac delta symbol which is equal to 1 if and only if $k_1 = k_2$ and 0 otherwise. We also need the operators $[[\partial_x^2]]$, $[[\partial_y^2]]$, $y[[\partial_x]]$ and $x[[\partial_y]]$ which are applied to the approximation $\widetilde{\phi}$ of ϕ, for $(k_1,k_2) \in \mathcal{O}_M$,

$$\begin{aligned}
\partial_x^2\phi(\mathbf{x}_{k_1,k_2}) &\approx ([[\partial_x^2]]\widetilde{\phi})_{k_1,k_2} := -\frac{1}{M}\sum_{p=-M/2}^{M/2-1} \xi_p^2 \widehat{(\widetilde{\phi}_{k_2})}_p e^{i\xi_p(x_{k_1}+L)},\\
\partial_y^2\phi(\mathbf{x}_{k_1,k_2}) &\approx ([[\partial_y^2]]\widetilde{\phi})_{k_1,k_2} := -\frac{1}{M}\sum_{q=-M/2}^{M/2-1} \mu_q^2 \widehat{(\widetilde{\phi}_{k_1})}_q e^{i\mu_q(y_{k_2}+L)},\\
(x\partial_y\phi)(\mathbf{x}_{k_1,k_2}) &\approx (x[[\partial_y]]\widetilde{\phi})_{k_1,k_2} := \frac{1}{M}\sum_{q=-M/2}^{M/2-1} ix_{k_1}\mu_q \widehat{(\widetilde{\phi}_{k_1})}_q e^{i\mu_q(y_{k_2}+L)},\\
(y\partial_x\phi)(\mathbf{x}_{k_1,k_2}) &\approx (y[[\partial_x]]\widetilde{\phi})_{k_1,k_2} := \frac{1}{M}\sum_{p=-M/2}^{M/2-1} iy_{k_2}\xi_p \widehat{(\widetilde{\phi}_{k_2})}_p e^{i\xi_p(x_{k_1}+L)}.
\end{aligned} \tag{3.1}$$

By considering the operators from $\mathbb{C}^N$ ($N = M^2$ (in 2D)) to $\mathbb{C}$ given by $[[\Delta]] := [[\partial_x^2]] + [[\partial_y^2]]$ and $[[\mathbb{L}_z]] := -i(x[[\partial_y]] - y[[\partial_x]])$, we obtain the discretization of the gradient of the energy

$$\nabla E(\phi) = 2H_\phi\phi, \quad \text{with } H_\phi = -\frac{1}{2}[[\Delta]] + [[V]] + \eta[[|\phi|^2]] - \omega[[L_z]].$$

We set $\phi := (\tilde{\phi}(\mathbf{x}_{k_1,k_2}))_{(k_1,k_2)\in\mathcal{O}_M}$ as the discrete unknown vector in $\mathbb{C}^N$. For conciseness, we identify an array ϕ in the vector space of 2D complex-valued arrays $\mathcal{M}_M(\mathbb{C})$ (storage according to the 2D grid) and the reshaped vector in $\mathbb{C}^N$. Finally, the cost for evaluating the application of a 2D FFT is $\mathcal{O}(N\log N)$.

In this discretization, the computations in Section 2 are still valid, with the following differences: ϕ is an element of $\mathbb{C}^N$, the operators Δ, V, $|\phi|^2$ and L_z are $N \times N$ Hermitian matrices, and the inner product is the standard $\mathbb{C}^N$ inner product. In the following sections, we will assume a discretization like the above one, and drop the brackets in the operators for conciseness.

4. Review and analysis of classical methods for computing the ground states of GPEs

Once an appropriate discretization is chosen, it remains to compute the solution to the discrete minimization problem

$$\phi \in \operatorname*{arg\,min}_{\phi\in\mathbb{C}^N,\|\phi\|=1} E(\phi). \tag{4.1}$$

Classical methods used to find solutions of (4.1) mainly use the so-called *imaginary time equation*, which is formally obtained by considering imaginary times in the time-dependent Schrödinger equation. Mathematically, this corresponds to the gradient flow associated with the energy E on the manifold $\mathcal{S}$:

$$\partial_t\phi = -\frac{1}{2}M_\phi\nabla E(\phi) = -(H_\phi\phi - \lambda(\phi)\phi). \tag{4.2}$$

As is well-known, the oscillatory behavior of the eigenmodes of the Schrödinger equation become dampening in imaginary time, thus decreasing the energy. The presence of the Lagrange multiplier λ, coming from the projection on the tangent space on $\mathcal{S}$, ensures the conservation of norm: $\|\phi\| = 1$ for all times. This equation can be discretized in time and solved. However, since H_ϕ is an unbounded operator, explicit methods encounter Courant–Friedrichs–Lewy (CFL) like conditions [27] that limit the size of their time step, and many authors [7,6,13–15,49] use a backward-Euler discretization scheme.

We are interested in this section in obtaining the asymptotic convergence rates of various discretizations of this equation, to compare different approaches. To this end, consider here the model problem of finding the first eigenpair of a $N \times N$ symmetric matrix H. We label its eigenvalues $\lambda_1 \le \lambda_2 \cdots \le \lambda_N$, and assume that the first eigenvalue λ_1 is simple, so that $\lambda_1 < \lambda_2$. This model problem is a linearized version of the full nonlinear problem. It is instructive for two reasons: first, any good algorithm for the nonlinear problem must also be a good algorithm for this simplified problem. Second, this model problem allows for a detailed analysis that leads to tractable convergence rates. These convergence rates allow a comparison between different schemes, and are relevant in the asymptotic regime of the nonlinear problem.

We consider the following discretizations of equation (4.2): Forward Euler (FE), Backward Euler (BE), and Crank–Nicolson (CN) schemes:

$$\frac{\tilde{\phi}_{n+1}^{\mathrm{FE}} - \phi_n}{\Delta t} = -(H\phi_n - \lambda(\phi_n)\phi_n), \tag{4.3}$$

$$\frac{\tilde{\phi}_{n+1}^{\mathrm{BE}} - \phi_n}{\Delta t} = -(H\tilde{\phi}_{n+1}^{\mathrm{BE}} - \lambda(\phi_n)\phi_n), \tag{4.4}$$

$$\frac{\tilde{\phi}_{n+1}^{\mathrm{CN}} - \phi_n}{\Delta t} = -\frac{1}{2}(H\tilde{\phi}_{n+1}^{\mathrm{CN}} - \lambda(\phi_n)\tilde{\phi}_{n+1}^{\mathrm{CN}}) - \frac{1}{2}(H\phi_n - \lambda(\phi_n)\phi_n). \tag{4.5}$$

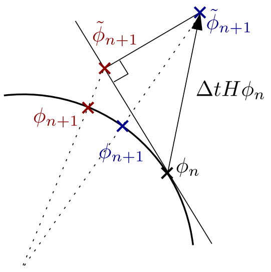

Fig. 1. Update rule with the projected gradient, with the λ term (red) and with the unprojected gradient, without the λ term (blue). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

These discretizations all decrease the energy when $\Delta t > 0$ is small enough, but do not preserve the norm: the departure from normalization is of order $\mathcal{O}(\Delta t^2)$. Therefore, they are followed by a projection step

$$\phi_{n+1} = \frac{\tilde{\phi}_{n+1}}{\left\|\tilde{\phi}_{n+1}\right\|}.$$

Note that some authors do not include the λ term, choosing instead to work with

$$\frac{\tilde{\phi}^{\mathrm{FE}}_{n+1} - \phi_n}{\Delta t} = -H\phi_n, \tag{4.6}$$

$$\frac{\tilde{\phi}^{\mathrm{BE}}_{n+1} - \phi_n}{\Delta t} = -H\tilde{\phi}^{\mathrm{BE}}_{n+1}, \tag{4.7}$$

$$\frac{\tilde{\phi}^{\mathrm{CN}}_{n+1} - \phi_n}{\Delta t} = -\frac{1}{2}(H\tilde{\phi}^{\mathrm{CN}}_{n+1} + H\phi_n). \tag{4.8}$$

These also yield schemes that decrease the energy. However, because of the use of the unprojected gradient $H\phi$ instead of $M_\phi(H\phi) = H\phi - \lambda\phi$, the departure from normalization is $\mathcal{O}(\Delta t)$, instead of $\mathcal{O}(\Delta t^2)$ for the projected case. The difference between the two approaches is illustrated in Fig. 1.

For the FE and BE methods, a simple algebraic manipulation shows that one step of the method with the λ term is equivalent to one step of the method without the λ term, but with an effective Δt modified as

$$\Delta_t^{\lambda,\mathrm{FE}} = \frac{\Delta t}{1 + \Delta t\lambda}, \tag{4.9}$$

$$\Delta_t^{\lambda,\mathrm{BE}} = \frac{\Delta t}{1 - \Delta t\lambda}. \tag{4.10}$$

This is not true for the CN method, nor it is true when nonlinear terms are included. However, even in this case, the difference between including and not including the λ term is $\mathcal{O}(\Delta t^2)$, and their behavior is similar. Since the analysis of unprojected gradient methods is simpler, we focus on this here.

Then, these schemes can all be written in the form

$$\phi_{n+1} = \frac{A\phi_n}{\|A\phi_n\|}, \tag{4.11}$$

where the matrix A is given by

$$A^{\mathrm{FE}} = I - \Delta t H,$$
$$A^{\mathrm{BE}} = (I + \Delta t H)^{-1},$$
$$A^{\mathrm{CN}} = (I + \frac{\Delta t}{2}H)^{-1}(I - \frac{\Delta t}{2}H).$$

The eigenvalues μ_i of A are related to the eigenvalues λ_i of H by the following spectral transformation

$$\mu_i^{\mathrm{FE}} = 1 - \Delta t\lambda_i,$$
$$\mu_i^{\mathrm{BE}} = \frac{1}{1 + \Delta t\lambda_i},$$
$$\mu_i^{\mathrm{CI}} = \frac{1 - \frac{\Delta t}{2}\lambda_i}{1 + \frac{\Delta t}{2}\lambda_i}.$$

We call this eigenvalue the amplification factor: if ϕ_n has eigencomponents $c_{n,i} = \langle v_i, \phi_n \rangle$ on the eigenvector v_i of H and ϕ_0 is normalized to 1, then the iteration (4.11) can be solved as

$$c_{n,i} = \frac{\mu_i^n}{\sqrt{\sum_{i=1}^N |\mu_i^n c_{0,i}|^2}} c_{0,i}.$$

This iteration converges towards the eigenvector associated to the largest eigenvalue μ_N (in modulus) of A, if it is simple, with convergence rate μ_{N-1}/μ_N. This is nothing but the classical power method for the computation of eigenvalues, with a spectral transformation from H to A. Therefore we identify the FE method as a shifted power method, the BE method as a shift-and-invert approach, and the CN uses a generalized Cayley transform [10,43].

From this we can readily see the properties of different schemes. We make the assumption that either λ_1 is positive or that $\Delta t < \frac{1}{-\lambda_1}$ (for BE) and $\Delta t < \frac{2}{-\lambda_1}$ (for CN). If this condition is not verified, then the iteration will generally not converge towards an eigenvector associated with λ_1 because another eigenvalue than λ_1 will have a larger amplification factor. Under this assumption, we see that the BE and CN converge unconditionally, while FE only converges if

$$\Delta t < \frac{2}{\lambda_N}.$$

This is a CFL-like condition: when H is the discretization of an elliptic operator, λ_N will tend to infinity as the size of the basis increases, which will force FE to take smaller time steps.

The asymptotic convergence rate of these methods is $\frac{\mu_{N-1}}{\mu_N}$. While the FE method has a bounded convergence rate, imposed by λ_1, λ_2 and λ_N, the BE and CN methods can be made to have an arbitrarily small convergence rate, by simply choosing Δt arbitrarily close to $-\frac{1}{\lambda_1}$ (BE) or $-\frac{2}{\lambda_1}$ (CN). Since in practice λ_1 is unknown, it has to be approximated, for instance by $\lambda(\phi_n)$. This yields the classical Rayleigh Quotient Iteration:

$$\phi_{n+1} = \frac{(H - \langle \phi_n, H\phi_n \rangle)^{-1}\phi_n}{\left\| (H - \langle \phi_n, H\phi_n \rangle)^{-1}\phi_n \right\|},$$

which is known to converge cubically. This iteration can also be seen as a Newton-like method.

From the previous considerations, it would seem that the BE and CN are infinitely superior to the FE method: even with a fixed stepsize, the BE and CN methods are immune to CFL-like conditions, and with an appropriately chosen stepsize, it can be turned into a superlinearly-convergent scheme. The first difficulty with this approach is that it is a linear strategy, only guaranteed to converge when close to the ground state. As is always the case with Newton-like methods, it requires a globalization strategy to be efficient and robust in the nonlinear setting. The second issue, is that the BE and CN methods require the solution of the linear system $(H - \lambda(\phi_n))\tilde{\phi}_{n+1} = \phi_n$.

The difficulty of solving the system depends on the discretization scheme used. For localized basis schemes like the finite element method, H is sparse, and efficient direct methods for large scale sparse matrices can be used [42]. For the Fourier pseudo-spectral scheme, which we use in this paper, H is not sparse, and only matrix–vector products are available efficiently through FFTs (a *matrix-free problem*). This means that the system has to be solved using an iterative method. Since it is a symmetric but indefinite problem, the solver of choice is MINRES [18], although the solver BICGSTAB has been used [7,9]. The number of iterations of this solver will then grow when the grid spacing tends to zero, which shows that BE also has a CFL-like limitation. However, as is well-known, Krylov methods [42] only depend on the square root of the condition number for their convergence, as opposed to the condition number itself for fixed-point type methods [15,49]. This explains why BE with a Krylov solver is preferred to FE in practice [7,9].

Furthermore, preconditioners can be used to reduce this number of iterations [7,9], e.g. with a simple preconditioner (one that is diagonal either in real or in Fourier space). This method is effective for many problems, but requires a globalization strategy, as well as an appropriate selection of parameters such as Δt and the precision used to solve the linear system [7,9]. Here, we propose a method that has all the advantages of BE (robust, Krylov-like dependence on the square root of the condition number, ability to use a preconditioner), but is explicit, converges faster than BE, and has no free parameter (no fine-tuning is necessary).

5. The preconditioned steepest descent (PSD) and conjugate gradient (PCG) methods

5.1. The steepest descent method

The previous approaches usually employed in the literature to compute the ground states of the Gross–Pitaevskii equation are all based on implicit discretizations of the imaginary-time equation (4.2). As such, these methods come from PDE theory and lack the insight of minimization algorithms. The difficulty of applying classical minimization algorithms comes from the spherical constraints. However, the general theory of optimization algorithms on Riemannian manifolds has been developed extensively in [2,33], where the authors derive constrained analogues of gradient, conjugate gradient and Newton's algorithms. This is the approach we follow here, and employ a preconditioned conjugate gradient method on the manifold $\mathcal{S}$.

In this section, we work with an arbitrary symmetric positive definite preconditioner P. The choice of P will be discussed later in subsection 5.5. The (projected, preconditioned) steepest descent method for the minimization of E on $\mathcal{S}$ is the update

$$\tilde{\phi}_{n+1} = \phi_n - \alpha_n P\left(H_{\phi_n}\phi_n - \lambda_n\phi_n\right), \quad \phi_{n+1} = \tilde{\phi}_{n+1} / \left\|\tilde{\phi}_{n+1}\right\|, \tag{5.1}$$

where $\lambda_n = \lambda(\phi_n)$. We reformulate this equation as

$$\phi_{n+1} = \cos(\theta_n)\phi_n + \sin(\theta_n)\frac{p_n}{\|p_n\|}, \quad \text{with} \quad p_n = d_n - \mathrm{Re}\,\langle d_n, \phi_n\rangle\,\phi_n, \tag{5.2}$$

where $d_n = -Pr_n$ is the descent direction, equal to the negative of the preconditioned residual $r_n = H_{\phi_n}\phi_n - \lambda(\phi_n)\phi_n$. The equations (5.1) and (5.2) are equivalent when θ_n or α_n is small enough, with a one-to-one correspondence between θ_n and α_n. To first order, we have: $\alpha_n = \theta_n\,\|p_n\|$.

Without preconditioner, this method, summarized in Algorithm 1, is identical to the FE method (4.6).

Algorithm 1: The steepest descent method.

```
while not converged do
    $\lambda_n = \lambda(\phi_n)$
    $r_n = H_{\phi_n}\phi_n - \lambda_n\phi_n$
    $d_n = -Pr_n$
    $p_n = d_n - \mathrm{Re}\,\langle d_n, \phi_n\rangle\,\phi_n$
    $\theta_n = \arg\min_\theta E\left(\cos(\theta)\phi_n + \sin(\theta)p_n/\|p_n\|\right)$
    $\phi_{n+1} = \cos(\theta_n)\phi_n + \sin(\theta_n)p_n/\|p_n\|$
    $n = n+1$
end
```

To choose the parameter θ_n, a number of strategies are possible. We first show that, when θ_n is small enough, the steepest descent method decreases the energy.

Expanding ϕ_{n+1} up to second-order in θ_n, we obtain

$$\phi_{n+1} = \left(1 - \frac{\theta_n^2}{2}\right)\phi_n + \theta_n\frac{p_n}{\|p_n\|} + \mathcal{O}(\theta_n^3), \tag{5.3}$$

and therefore

$$E(\phi_{n+1}) = E(\phi_n) + \frac{\theta_n}{\|p_n\|}\mathrm{Re}\,\langle\nabla E(\phi_n), p_n\rangle + \frac{1}{2}\frac{\theta_n^2}{\|p_n\|^2}\left[\nabla^2 E(\phi_n)[p_n, p_n] - \lambda_n\|p_n\|^2\right] + \mathcal{O}(\theta_n^3). \tag{5.4}$$

We now compute the first-order variation

$$\begin{aligned}\mathrm{Re}\,\langle\nabla E(\phi_n), p_n\rangle &= \frac{\mathrm{Re}\,\langle\nabla E(\phi_n), d_n - \mathrm{Re}\,\langle d_n, \phi_n\rangle\,\phi_n\rangle}{\|d_n - \mathrm{Re}\,\langle d_n, \phi_n\rangle\,\phi_n\|} = \frac{\mathrm{Re}\,\langle r_n, d_n\rangle}{\|d_n - \mathrm{Re}\,\langle d_n, \phi_n\rangle\,\phi_n\|}\\ &= -\frac{\langle r_n, Pr_n\rangle}{\|d_n - \mathrm{Re}\,\langle d_n, \phi_n\rangle\,\phi_n\|}.\end{aligned}$$

Since P was assumed to be positive definite, this term is always negative so that the algorithm decreases the energy when θ_n is chosen small enough. Since p_n is orthogonal to ϕ_n, the second-order term $\nabla^2 E(\phi_n)[p_n, p_n] - \lambda_n\|p_n\|^2$ is guaranteed to be positive when ϕ_n is close to a minimizer by the second-order optimality conditions.

Therefore, a basic strategy is to choose θ_n fixed and small enough so that the energy decreases. A better one is to choose θ_n adaptively. For instance, we could perform the linesearch

$$\theta_n = \arg\min_\theta E\left(\cos(\theta)\phi_n + \sin(\theta)\frac{p_n}{\|p_n\|}\right). \tag{5.5}$$

Since $E(\theta)$ is not a quadratic function, this is a nonlinear one-dimensional minimization problem, generally requiring many evaluations of $E(\theta)$ to converge to a minimum. However, many of the computations for the evaluation of $E(\theta)$, including all that require FFTs, can be pre-computed. Since the FFT step is the dominant one in the computation of the energy, the evaluation of $E(\theta)$ at many points is not much more costly than the evaluation at a single point. Therefore it is feasible to use a standard one-dimensional minimization routine.

Alternatively, we can obtain a simple and cheap approximation by minimizing the second-order expansion of E in θ_n. We expect this to be accurate when θ_n is small, which is the case close to a minimizer. Minimizing (5.4) with respect to θ_n yields

$$\theta_n^{\mathrm{opt}} = \frac{-\mathrm{Re}\,\langle\nabla E(\phi_n), p_n\rangle\,\|p_n\|}{\mathrm{Re}\left[\nabla^2 E(\phi_n)[p_n, p_n] - \lambda_n\right]}. \tag{5.6}$$

As we have seen, the numerator is always positive, and the denominator is positive when ϕ_n is close enough to a minimizer. In our implementation, we compute the denominator, and, if it is positive, we use θ_n^{opt} as a trial stepsize. If not, we use some default positive value. If the energy of ϕ_{n+1} using this trial stepsize is decreased, we accept the step. If the energy is not decreased, we reject the step, decrease the trial stepsize, and try again, until the energy is decreased (which is mathematically ensured when θ_n is small enough). Alternatively, we can use Armijo or Wolfe conditions as criterion to accept or reject the stepsize, or even use the full line search (5.5). The evaluation of the energy at multiple values of θ do not require more Fourier transforms than at only one point, but only more computations of the nonlinear term, so that a full line search is not much more costly than the heuristic outlined above. In our tests however, the heuristic above was sufficient to ensure fast convergence, and a full line search only marginally improved the number of iterations. Therefore, we simply use the heuristic in the numerical results of Section 6.

Let us note that under reasonable assumptions on the structure of critical points and on the stepsize choice, there are various results on the convergence of this iteration to a critical point (see [2] and references therein).

5.2. The conjugate gradient method

The conjugate gradient method is very similar, but uses an update rule of the form

$$d_n = -Pr_n + \beta_n p_{n-1} \tag{5.7}$$

instead of simply $d_n = -Pr_n$. This is justified when minimizing unconstrained quadratic functionals, where the formula

$$d_n = -Pr_n + \beta_n d_{n-1}, \quad \text{with} \quad \beta_n = \frac{\langle r_n, Pr_n\rangle}{\langle r_{n-1}, Pr_{n-1}\rangle}, \tag{5.8}$$

yields the well-known PCG method to solve linear systems. For nonlinear problems, different update formulas can be used, all equivalent in the linear case. Equation (5.8) is known as the Fletcher–Reeves update. Another popular formula is the Polak–Ribière choice $\beta = \max(\beta^{\mathrm{PR}}, 0)$, where

$$\beta^{\mathrm{PR}} = \frac{\langle r_n - r_{n-1}, Pr_n\rangle}{\langle r_{n-1}, Pr_{n-1}\rangle}. \tag{5.9}$$

We use $\beta = \max(\beta^{\mathrm{PR}}, 0)$, which is equivalent to restarting the CG method (simply using a steepest descent step) when $\beta^{\mathrm{PR}} < 0$ and is a standard choice in nonlinear CG methods. For the justification of the CG method for constrained minimization, see [2,33].

Algorithm 2: The conjugate gradient method.

```
while not converged do
    $\lambda_n = \lambda(\phi_n)$
    $r_n = H_{\phi_n}\phi_n - \lambda_n\phi_n$
    $\beta_n = \langle r_n - r_{n-1}, Pr_n\rangle / \langle r_{n-1}, Pr_{n-1}\rangle$
    $\beta_n = \max(\beta_n, 0)$
    $d_n = -Pr_n + \beta p_{n-1}$
    $p_n = d_n - \mathrm{Re}\,\langle d_n, \phi_n\rangle\,\phi_n$
    $\theta_n = \arg\min_\theta E\,(\cos(\theta)\phi_n + \sin(\theta)p_n/\|p_n\|)$
    $\phi_{n+1} = \cos(\theta_n)\phi_n + \sin(\theta_n)p_n/\|p_n\|$
    $n = n + 1$
end
```

The CG algorithm is presented in Algorithm 2. In contrast with the steepest descent algorithm, the quantity $\mathrm{Re}\,\langle\nabla E(\phi_n), p_n\rangle$ does not have to be negative, and p_n might not be a descent direction: even with a small stepsize, the energy does not have to decrease at each step. To obtain a robust minimization method, we enforce energy decrease to guarantee convergence. Therefore, our strategy is to first check if p_n is a descent direction by computing $\mathrm{Re}\,\langle\nabla E(\phi_n), p_n\rangle$. If p_n is not a descent direction, we revert to a steepest descent step, which we know will decrease the energy, else, we choose θ_n as in (5.6), and use the same step size control as in the steepest descent algorithm.

In our numerical tests, we observe that these precautions of checking the descent direction and using a stepsize control mechanism are useful in the first stage of locating the neighborhood of a minimum. Once a minimum is approximately located, p_n is always a descent direction and the stepsize choice (5.6) always decreases the energy.

5.3. Stopping criteria

A common way to terminate the iteration (in the BE schemes) is to use the stopping criterion

$$\phi_{\mathrm{err}}^{n,\infty} := \|\phi_{n+1} - \phi_n\|_\infty \le \varepsilon. \tag{5.10}$$

This can be problematic because the minima are generally not isolated but form a continuum due to symmetries (for instance, complex phase or rotational invariance), and this criterion might be too restrictive. A more robust one is based on the norm of the symmetry-covariant residual

$$r_{\text{err}}^{n,\infty} := \|r_n\|_\infty = \|H_{\phi_n}\phi_n - \lambda_n\phi_n\|_\infty \leq \varepsilon, \tag{5.11}$$

or the symmetry-invariant energy difference

$$\mathcal{E}_{\text{err}}^{n} := |E(\phi_{n+1}) - E(\phi_n)| \leq \varepsilon. \tag{5.12}$$

This third one converges more rapidly than the two previous ones: as is standard in optimization, when ϕ^* is a minimum of the constrained minimization problem and $\phi \in \mathcal{S}$, then

$$E(\phi) - E(\phi^*) = \mathcal{O}(\|\phi - \phi^*\|^2).$$

This is consistent with our results in Fig. 7.

In the current paper, we always use the energy based stopping criterion (5.12): for the 2D and 3D cases, a criteria based on $\phi_{\text{err}}^{n,\infty}$ or $r_{\text{err}}^{n,\infty}$ can lead to long computational times, most particularly for large rotations ω, even without changing the energy (see the example in subsection 6.2).

5.4. Convergence analysis

A full analysis of the convergence properties of our methods is beyond the scope of this paper, but we give in this section some elementary properties, and heuristics to understand their asymptotic performance.

Based on the expansion of the energy (5.4) as a function of θ for the steepest descent method, it is straightforward to prove that, when E is bounded from below and the step size θ is chosen optimally, the norm of the projected gradient $H_{\phi_n}\phi_n - \lambda_n\phi_n$ converges to 0. Convergence guarantees for the conjugate gradient method are harder, but can still be proven under a suitable restart strategy that ensures that the energy always decreases fast enough (for instance, the Armijo rule).

With additional assumptions on the non-degeneracy of critical points, we can even prove the convergence of ϕ_n to a critical point, that will generically be a local minimum. However, the question of the precise convergence speed of the steepest descent and conjugate gradient algorithms we use is problematic, because of three factors: the constraint $\|\phi\| = 1$, the non-quadraticity of E, and the presence of a preconditioner. To our knowledge, no asymptotically optimal bound for this problem has been derived. Nevertheless, based on known results about the convergence properties of the conjugate gradient method for preconditioned linear systems on the one hand [42], and of steepest descent methods for nonlinear constrained minimization [2] on the other hand, it seems reasonable to expect that the convergence will be influenced by the properties of the operator

$$M = (1 - \phi\phi^*)P(\nabla^2 E(\phi) - \lambda I)(1 - \phi\phi^*), \tag{5.13}$$

where ϕ is the minimum and $\lambda = \lambda(\phi)$. This operator admits 0 as its lowest eigenvalue, associated with the eigenvector ϕ. It is reasonable to expect that the convergence rate will be determined by a condition number σ equals to the ratio of the largest to the lowest non-zero eigenvalue of this operator. As is standard for linear systems, we also expect that the number of iterations to achieve a given tolerance will behave like $\sqrt{\sigma}$ for the conjugate gradient algorithm, and σ for the steepest descent algorithm. As we will see in Section 6, this is verified in our tests.

The Hessian operator $\nabla^2 E(\phi)$, which includes a Laplace operator, is not bounded. Correspondingly, on a given discretization domain, when the grid is refined, the largest eigenvalues of this operator will tend to $+\infty$. For a linear meshsize h, the eigenvalues of $\nabla^2 E(\phi)$ will behave as $\mathcal{O}(h^{-2})$. This is another instance of the CFL condition already seen in the discretization of the imaginary time equation. The Hessian $\nabla^2 E$ also includes a potential term V, which is often confining and therefore not bounded, such as the classical harmonic potential $V(\mathbf{x}) = |\mathbf{x}|^2$, or more generally confining potentials whose growth at infinity is like $|\mathbf{x}|^p$, for some $p > 0$. Thus, even with a fixed meshsize h on a domain $[-L, L]^d$, when L is increased, so will the largest eigenvalues of $\nabla^2 E(\phi)$, with a $\mathcal{O}(L^p)$ growth. When a steepest descent or conjugate gradient method is used without preconditioning, the convergence will be dominated by modes associated with largest eigenvalues of $\nabla^2 E$. This appears in simulations as high-frequency oscillations and localization at the boundary of the domain of the residual r_n.

To remedy these problems and achieve a good convergence rate, adequate preconditioning is crucial.

5.5. Preconditioners

We consider the question of building preconditioners P for the algorithms presented above. In the schemes based on the discretization of the gradient flow, preconditioning is naturally needed when solving linear systems by iterative methods. In the steepest descent and conjugate gradient optimization schemes, it appears as a modification of the descent direction to make it point closer to the minimum:

$$d_n := -P(H_{\phi_n}\phi_n - \lambda_n\phi_n). \tag{5.14}$$

In both cases, the preconditioning matrix should be an approximation of the inverse of the Hessian matrix of the problem. If no preconditioner was applied, we denote

$$P_I = I, \tag{5.15}$$

where I is the identity operator.

Kinetic energy preconditioner. One of these approximations is to use only the kinetic energy term

$$P_\Delta = (\alpha_\Delta - \Delta/2)^{-1}, \tag{5.16}$$

where α_Δ is a positive shifting constant to get an invertible operator. This has been called a "Sobolev gradient" in [30] because it is equivalent to taking the gradient of the energy in the Sobolev H^1-norm (with $\alpha_\Delta = 1/2$). In the framework of the BESP scheme for the GPE with Krylov solver, a similar preconditioner has been proposed in [7], α_Δ being the inverse of the time step Δt of the semi-implicit Euler scheme. A closely-related variant is standard in plane-wave electronic structure computation [50], where it is known as the Tetter–Payne–Allan preconditioner [45]. This preconditioner is diagonal in Fourier space and can therefore be applied efficiently in our pseudo-spectral approximation scheme. On a fixed domain $[-L, L]^d$, the effect of this preconditioner is to make the number of iterations independent from the spatial resolution h, because $P\nabla^2 E(\phi)$, seen as an operator on the space of functions on $[-L, L]^d$, will be equal to the identity plus a compact operator. This is supported by numerical experiments in Section 6. However, this operator is not bounded in the full domain $\mathbb{R}^d$. Therefore, as L increases, so will the largest eigenvalues of $P\nabla^2 E(\phi)$. For a potential V that grows at infinity like $|\mathbf{x}|^p$, the largest eigenvalues of $P\nabla^2 E(\phi)$ are $\mathcal{O}(L^p)$, resulting in an inefficient preconditioner. Similarly, when η is large, the nonlinear term becomes dominant, and the kinetic energy preconditioner is inefficient.

The choice of α_Δ is a compromise: if α_Δ is too small, then the preconditioner will become close to indefinite, which can produce too small eigenvalues in the matrix (5.13) and hamper convergence. If α_Δ is too big, then the preconditioner does not act until very large frequencies, and large eigenvalues result. We found that a suitable adaptive choice, that has consistently good performance and avoids free parameters, is

$$\alpha_\Delta = \tilde{\lambda}_n := \int \left(\frac{1}{2}|\nabla\phi_n|^2 + V|\phi_n|^2 + \eta|\phi_n|^4\right) d\mathbf{x} > 0 \tag{5.17}$$

which is a positive number that represents the characteristic energy of ϕ_n. We use this choice for our numerical simulations.

Potential energy preconditioner. Another natural approach is to use the potential energy term for the preconditioner:

$$P_V = (\alpha_V + V + \eta|\phi_n|^2)^{-1}. \tag{5.18}$$

Similarly, α_V is a positive shifting constant to get an invertible operator. This preconditioner is diagonal in real space and can therefore be applied efficiently. Dual to the previous case, this preconditioner has a stable performance when the domain and η are increased, but deteriorates as the spatial resolution is increased. Such a preconditioner has been used in [7] when the gradient flow for the GPE is discretized through a BE scheme, leading then to a Thomas–Fermi preconditioner. In this study, the parameter α_V was $1/\Delta t$. As in the kinetic energy case, we found it efficient to use $\alpha_V = \tilde{\lambda}_n$, and we will only report convergence results for this choice of parameter.

Combined preconditioner. In an attempt to achieve a stable performance independent of the size of the domain or the spatial resolution, we can define the combined preconditioners

$$P_{C_1} = P_V P_\Delta, \quad P_{C_2} = P_\Delta P_V \tag{5.19}$$

or a symmetrized version

$$P_C = P_V^{1/2} P_\Delta P_V^{1/2}. \tag{5.20}$$

With these preconditioners, $P\nabla^2 E(\phi)$ is bounded as an operator on $L^2(\mathbb{R}^d)$ (this can be proven by writing explicitly its kernels in Fourier space and then using Schur's test). However, we found numerically that this operator is not bounded away from zero, and has small eigenvalues of size $\mathcal{O}(L^{-p} + h^2)$. Therefore, the conditioning deteriorates as both the spatial resolution and the size of the domain increase.

In summary, for a spatial resolution h and a domain size L, the asymptotic condition numbers of the preconditioned Hessian with these preconditioners are

$$\begin{aligned} &\kappa_\Delta = \mathcal{O}(L^p), \\ &\kappa_V = \mathcal{O}(h^{-2}), \\ &\kappa_C = \mathcal{O}\left(\tfrac{1}{L^{-p}+h^2}\right) = \mathcal{O}(\min(L^p, h^{-2})). \end{aligned} \tag{5.21}$$

Therefore, the combined preconditioners act asymptotically as the best of both the kinetic and potential preconditioners. However, they might not be more efficient in the pre-asymptotic regime and require additional Fourier transforms.

Computational efficiency. The application the operator P_V is almost free (since it only requires a scaling of ϕ), but the naive application of P_Δ requires a FFT/IFFT pair. However, since we apply the preconditioners after and before an application of the Hamiltonian, we can reuse FFT and IFFT computations, so that the application of P_Δ does not require any additional Fourier transform. Similarly, the use of P_{C_1} and P_{C_2} only require one additional Fourier transform per iteration, and that of the symmetrized version P_C two.

In summary, the cost in terms of Fourier transforms per iteration for the rotating GPE model is

- no preconditioner: 3 FFTs/iteration (get the Fourier transform of ϕ, and two IFFTs to compute $\Delta\phi$ and $L_z\phi$ respectively),
- P_Δ or P_V: 3 FFTs/iteration,
- non-symmetric combined P_{C_1} or P_{C_2}: 4 FFTs/iteration,
- symmetric combined P_C: 5 FFTs/iteration.

Note that this total cost might be different for another type of GPE model e.g. when a nonlocal dipole–dipole interaction is included [9,16].

As we will see in Section 6, all combined preconditioners have very similar performance, but the symmetrized one might be more stable in some circumstances. A theoretical explanation of these observations, and in particular of the effect of a non-symmetric preconditioner is, to the best of our knowledge, still missing.

6. Numerical results

We first introduce some notations. When we combine one of the preconditioners P_ν ($\nu = I$, Δ, V, C, C$_1$, C$_2$) (5.15)–(5.20) with the steepest descent algorithm (Algorithm 1), we denote the resulting methods by PSD$_\nu$. Similarly, we denote by PCG$_\nu$ if the preconditioned conjugate gradient algorithm (Algorithm 2) was applied. In the following, we denote by #iter the number of global iterations for an iterative algorithm to get the converged solution with an *a priori* tolerance ε with respect to the stopping criterion (5.12).

Concerning the BESP schemes (4.4) and (4.7), at each outer iteration n, one needs to solve an implicit system with the operator $(1/\Delta t + H_{\phi_n})$. We use a Krylov subspace iterative solver (MINRES here) with one of the preconditioners P_ν $(\nu = I, \Delta, V, \mathrm{C})$ (5.16)–(5.19) to accelerate the number of inner iterations [7]. The preconditioned BESP schemes are then denoted by BE$_\nu$, according to the chosen preconditioner. The number of iterations reported is equal to the sum of the inner iterations over the outer iterations.

In the following numerical experiments, we consider two types of trapping potential $V(\mathbf{x})$: the harmonic plus lattice potential [13]

$$V(\mathbf{x}) = V_d^0(\mathbf{x}) + \begin{cases} \kappa_x \sin^2(q_x x^2), \\ \sum_{\nu=x,y} \kappa_\nu \sin^2(q_\nu \nu^2), \\ \sum_{\nu=x,y,z} \kappa_\nu \sin^2(q_\nu \nu^2), \end{cases} \quad \text{with} \quad V_d^0(\mathbf{x}) = \begin{cases} \gamma_x^2 x^2, & d=1, \\ \sum_{\nu=x,y} \gamma_\nu \nu^2, & d=2, \\ \sum_{\nu=x,y,z} \gamma_\nu \nu^2, & d=3, \end{cases} \tag{6.1}$$

and the harmonic plus quartic potential for $d = 2, 3$ [29,30,49]

$$V(\mathbf{x}) = (1-\alpha) V_2^0(\mathbf{x}) + \frac{\kappa\,(x^2+y^2)^2}{4} + \begin{cases} 0, & d=2, \\ \gamma_z^2 z^2, & d=3. \end{cases} \tag{6.2}$$

Moreover, unless stated otherwise, we take the initial data as the Thomas Fermi approximation [7,13]:

$$\phi_0 = \frac{\phi_g^{\mathrm{TF}}}{\|\phi_g^{\mathrm{TF}}\|}, \qquad \text{with} \qquad \phi_g^{\mathrm{TF}}(\mathbf{x}) = \begin{cases} \sqrt{\left(\mu_g^{\mathrm{TF}} - V(\mathbf{x})\right)/\eta}, & V(\mathbf{x}) < \mu_g^{\mathrm{TF}}, \\ 0, & \text{otherwise}, \end{cases} \tag{6.3}$$

where

$$\mu_g^{\mathrm{TF}} = \frac{1}{2} \begin{cases} (3\eta\gamma_x)^{2/3}, & d=1, \\ (4\eta\gamma_x\gamma_y)^{1/2}, & d=2, \\ (15\eta\gamma_x\gamma_y\gamma_z)^{2/5}, & d=3. \end{cases} \tag{6.4}$$

The algorithms were implemented in Matlab (Release 8.5.0).

6.1. Numerical results in 1D

Here, $V(\mathbf{x})$ is chosen as the harmonic plus lattice potential (6.1) with $\gamma_x = 1$, $k_x = 25$ and $q_x = \frac{\pi}{2}$. The computational domain and mesh size are respectively denoted as $\mathcal{D} = [-L, L]$ and h. In addition, to compare with the common existing method BESP, we choose the stopping criteria (5.12) with $\varepsilon = 10^{-14}$ all through this section. For BESP, we choose $\Delta t = 0.01$ unless specified otherwise, and fix the error tolerance for the inner loop to 10^{-10}. Other values of the error tolerance were also tried, but this choice was found to be representative for the performance of BESP.

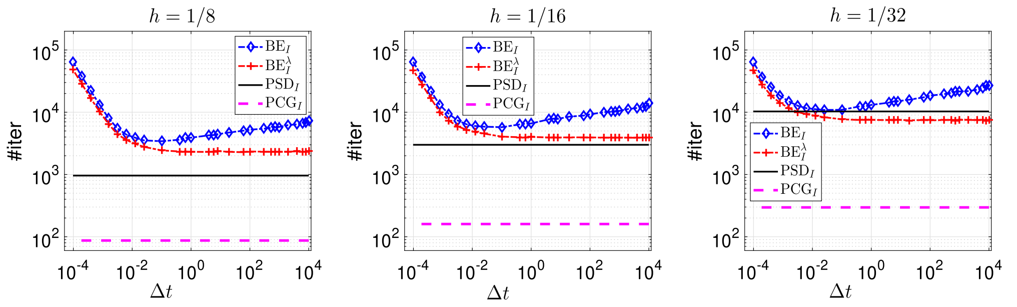

Fig. 2. Example 6.1. Number of iterations to converge for different methods and different stepsizes, with different values of the discretization parameter h.

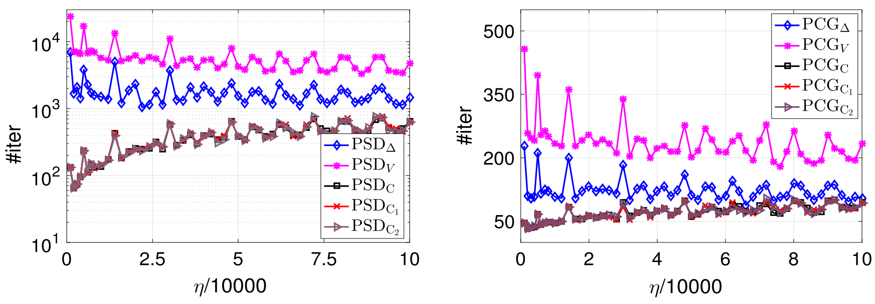

Fig. 3. Example 6.2. Number of iterations of PSD_ν and PCG_ν ($\nu = \Delta, V, \mathrm{C}, \mathrm{C}_1, \mathrm{C}_2$) to converge, for different nonlinear strengths η.

Example 6.1. We first compare the performance of various solvers without preconditioning in a simple case. We choose $L = 16$ and $\eta = 250$, and varying mesh sizes. We compare the method BE_I given by (4.7), BE_I^λ given by (4.4), and the steepest descent and conjugate gradient algorithms in Fig. 2. The difference between the methods BE_I and BE_I^λ is the inclusion of the chemical potential in the discretized gradient flow: we showed in Section 4 that both were equivalent for linear problems up to a renormalization in Δt. We see here that this conclusion approximately holds even in the nonlinear regime ($\eta \neq 0$), with both methods performing very similarly until Δt becomes large, at which point the BE_I^λ effectively uses a constant stepsize (see (4.10)), while the large timestep in BE_I makes the method inefficient. In this case, λ_1 is positive, so that both methods converge to the ground state even for a very large Δt. Overall we see that the optimum number of iterations is achieved for a value of Δt of about 0.01, which we keep in the following tests to ensure a fair comparison. We also use the BE^λ variant in the following tests.

For modest values of the discretization parameter h, the Backward Euler methods are less efficient than the steepest descent method (which can be interpreted as a Forward Euler iteration with adaptive stepsize). As h is decreased, the conditioning of the problem increases as h^{-2}. The steepest descent/Forward Euler method is limited by its CFL condition, and its number of iterations grows like h^{-2}, as can readily be checked in Fig. 2. The Backward Euler methods, however, use an efficient Krylov solver that is only sensitive to the square root of the conditioning, and its number of iterations grows only like h^{-1}. Therefore it become more efficient than the steepest descent/Forward Euler method.

The conjugate gradient method is always more efficient than the other methods by factors varying between one and two orders of magnitude. Its efficiency can be attributed to the combination of Krylov-like properties (as the Backward Euler method, its iteration count displays only a h^{-1} growth) and optimal stepsizes.

Example 6.2. We compare now the performance of the steepest descent and conjugate gradient methods with different preconditioners. To this end, we consider the algorithms PSD_ν and PCG_ν with $\nu = \Delta, V, \mathrm{C}, \mathrm{C}_1, \mathrm{C}_2$. The computational parameters are chosen as $L = 128$ and $h = \frac{1}{64}$, respectively. Fig. 3 shows the iteration number #iter for these schemes and different values of the nonlinearity strength η. From this figure and other numerical results not shown here, we can see that: (i) For each fixed preconditioner, the PCG schemes works better than the PSD schemes; (ii) the combined preconditioners all work equally well, and bring a reduction in the number of iteration, at the price of more Fourier transforms.

Example 6.3. In this example, we compare the performance of PSD_ν, PCG_ν and BE_ν ($\nu = I, \Delta, V, \mathrm{C}$) with respect to different domain and mesh sizes. To this end, we fix $\eta = 250$. Fig. 4 shows the total iteration number for these schemes with different L and h. From this figure and additional numerical results not shown here for brevity, we see that: (i) Preconditioned solvers outperform unpreconditioned solvers; (ii) The potential preconditioner P_V (5.18) makes the solver mainly depend on the spatial resolution h, while the kinetic potential preconditioner P_Δ (5.16) prevents the deterioration as h decreases for a fixed L, consistent with the theoretical analysis in subsection 5.5; (iii) The deterioration is less marked for Krylov-based methods (BE and PCG) than for the PSD method, because Krylov methods only depends on the square root of the condition

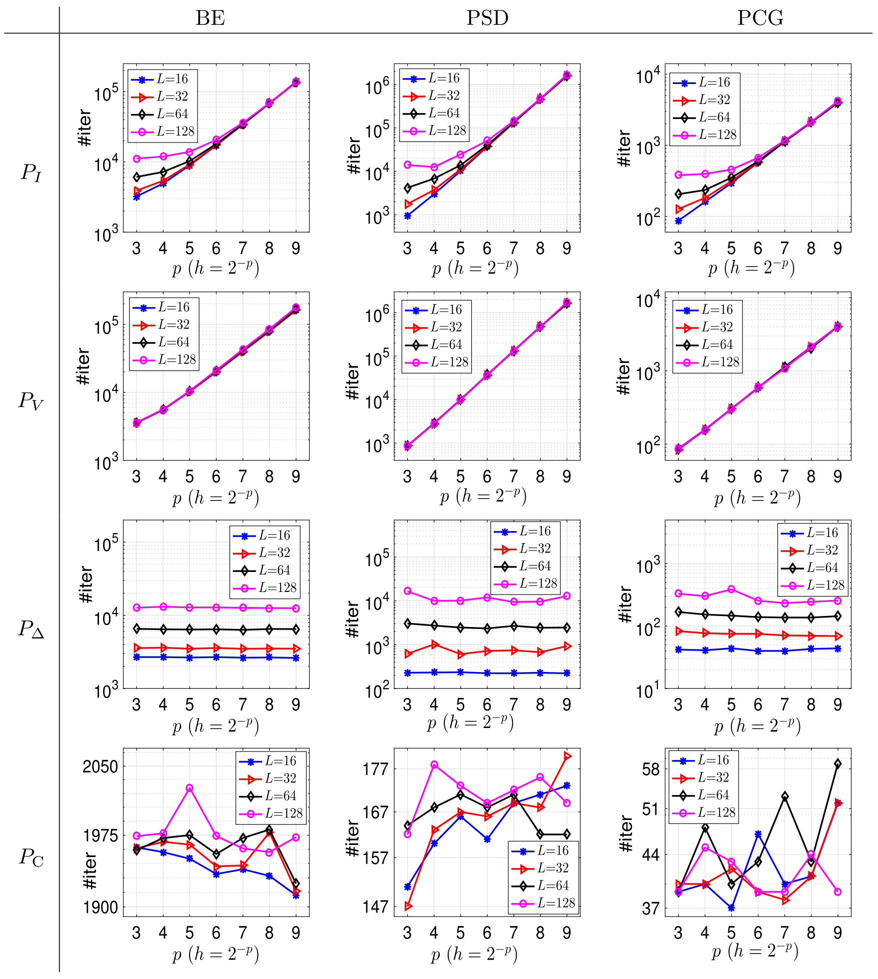

Fig. 4. Example 6.3. Number of iterations to converge for BE$_\nu$, PSD$_\nu$ and PCG$_\nu$ for $\nu = I, V, \Delta, \text{C}$, vs. the mesh refinement h.

number (iv) The combined preconditioner P_C (5.19) makes the solvers almost independent of both the parameters h and L, although we theoretically expect a stronger dependence. We attribute this to the fact that we start with a specific initial guess that does not excite the slowly convergent modes enough to see the dependence on h and L; (v) For each solver, the combined preconditioner P_C performs best. Usually, PCG$_\text{C}$ is the most efficient algorithm, followed by PSD$_\text{C}$, and finally BE$_\text{C}$.

Example 6.4. We investigate further the performance of PCG$_\text{C}$ and PSD$_\text{C}$ with different nonlinear interaction strenghts η. To this end, we take $L = 128$ and different discretization parameters h. We vary the nonlinearity from $\eta = 0$ to $\eta = 10^5$. Fig. 5 depicts the corresponding iteration numbers to converge. We could clearly see that: (i) The iteration counts for both methods are almost independent on h, but both depend on the nonlinearity η; PCG$_\text{C}$ depends slightly on η while PSD$_\text{C}$ is more sensitive; (ii) For fixed η and h, PCG$_\text{C}$ converges much faster than PSD$_\text{C}$.

From Examples 6.2–6.4, we see that PCG$_\text{C}$, i.e. PCG with combined symmetric preconditioner P_C (5.19) is the best solver. Hereafter, unless stated, we use PCG$_\text{C}$ as the default solver to compute the ground states.

6.2. Numerical results in 2D

Here, we choose $V(\mathbf{x})$ as the harmonic plus quartic potential (6.2) with $\gamma_x = \gamma_y = 1$, $\alpha = 1.2$ and $\kappa = 0.3$. The computational domain and mesh sizes are chosen respectively as $\mathcal{D} = [-16, 16]^2$ and $h = \frac{1}{16}$.

First, we test the evolution of the three errors (5.10)–(5.12) as the algorithm progresses. To this end, we take $\eta = 1000$ and $\omega = 3.5$ as example. Fig. 6 plots the $\phi_\text{err}^{n,\infty}$, $r_\text{err}^{n,\infty}$ and $\mathcal{E}_\text{err}^n$ errors with respect to the iteration number. We can see clearly that $\mathcal{E}_\text{err}^n$ converges faster the other two indicators, as expected. Considering $\phi_\text{err}^{n,\infty}$ or $r_\text{err}^{n,\infty}$ with an improper but relatively large tolerance would require a very long computational time to converge even if the energy would not change so much.

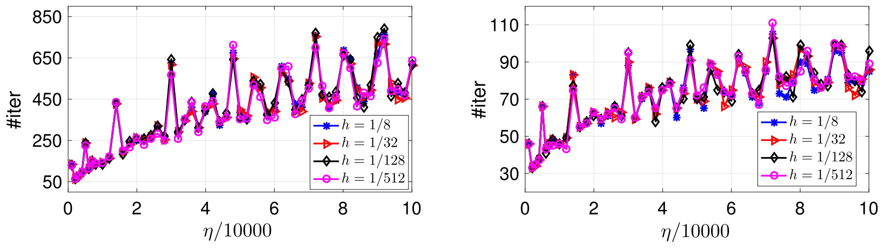

Fig. 5. Example 6.4. Number of iterations to converge for PSD_C (left) and PCG_C (right) with $L = 128$, and various values of h and η.

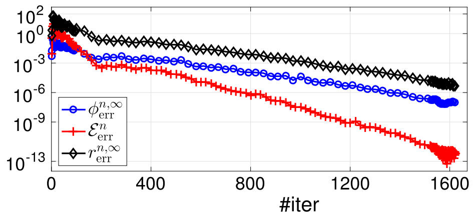

Fig. 6. Evolution of the errors vs. the total number of iterations.

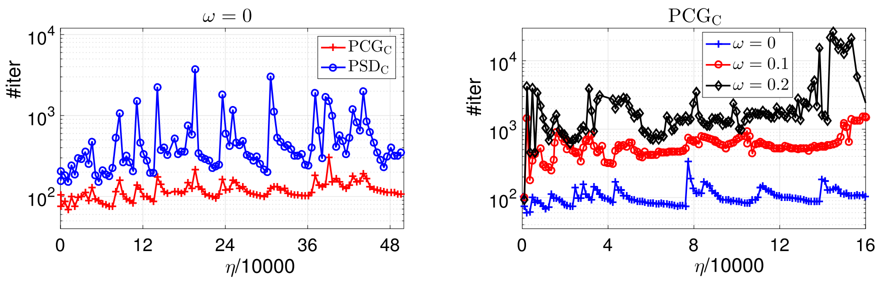

Fig. 7. Example 6.5. Number of iterations for PCG_C and PSD_C for $\omega = 0$ (left) and PCG_C for different ω (right) vs. η.

This is most particularly true for large values of ω. In all the examples below, unless otherwise stated, we fix $\mathcal{E}^n_{\mathrm{err}}$ (5.12) with $\varepsilon = 10^{-12}$ to terminate the code.

Example 6.5. In this example, we compare the performance of PCG_C and PSD_C for the 2D rotating case. To this end, $V(\mathbf{x})$ is chosen as the harmonic plus lattice potential (6.1) with $\gamma_x = \gamma_y = 1$, $k_x = k_y = 25$ and $q_x = q_y = \frac{\pi}{2}$. The computational domain and mesh sizes are chosen respectively as $\mathcal{D} = [-32, 32]^2$ and $h = \frac{1}{8}$. Fig. 7 (left) shows the iteration number of PCG_C and PSD_C vs. different values of η for $\omega = 0$, while Fig. 7 (right) reports the number of iterations of PCG_C with respect to η and ω. From this figure, we can see that: (i) Similarly to the 1D case, PCG_C outperforms PSD_C; (ii) For $\omega = 0$, the iteration number for PCG_C would oscillate in a small regime, which indicates the very slight dependence with respect to the nonlinearity strength η. When ω increases, the number of iterations increases for a fixed η. Meanwhile, the dependency on η becomes stronger as ω increases. Let us remark here that it would be extremely interesting to build a robust preconditioner including the rotational effects to get a weaker ω-dependence in terms of convergence.

Example 6.6. Following the previous example, here we compare the performance of PCG_C and PCG_{C_1} for different values ω. To this end, we fix $\eta = 1000$ and vary ω from 0 to 3.5. Fig. 8 illustrates the number of iterations of these method vs. different values of ω and there corresponding energies. From this figure and other experiments now shown here, we see that (i) All the methods converge to the stationary state with same energy; (ii) The symmetrized preconditioner has more stable performance than the non-symmetric version, a fact we do not theoretically understand.

Example 6.7. In this example, we apply PCG_C to solve some more difficult problems. We compute the ground states ϕ_g of rotating BECs with large values of η and ω. To this end, we take $L = 20$, $h = 1/16$ and set the stopping tolerance in (5.12) to $\varepsilon = 10^{-14}$. Table 1 lists the CPU times for the PCG_C solver to converge while Fig. 9 shows the contour plot of the density function $|\phi_g(\mathbf{x})|^2$ for different ω and η. We can see that the PCG_C method converges very fast to the stationary states. Let us remark that, to the best of our knowledge, only a few results were reported for such fast rotating BECs with highly nonlinear (very large η) problems, although they are actually more relevant for real physical problems. Hence, PCG_C can tackle efficiently difficult realistic problems on a laptop.

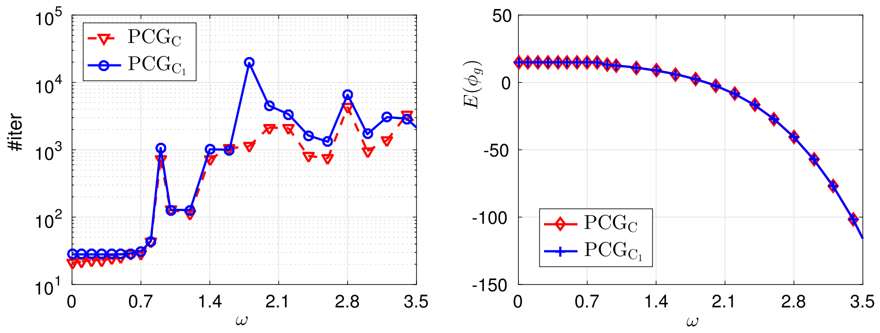

Fig. 8. Example 6.6. Number of iterations for PCG_C and PCG_{C_1} (left) and its corresponding total energies (right) vs. ω.

Table 1
CPUs time (*seconds*) for PCG_C to compute the ground states of the GPE with various ω and η.

η	$\omega = 1$	1.5	2	2.5	3	3.5	4	4.5
1000	493	551	560	2892	2337	720	966	3249
5000	1006	1706	867	6023	1144	1526	12514	19248
10000	4347	21525	5511	15913	15909	6340	16804	32583

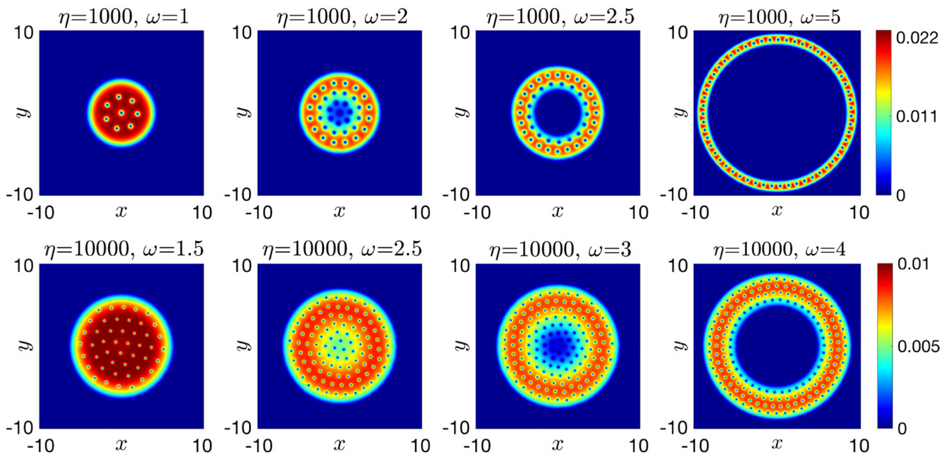

Fig. 9. Example 6.7. Corresponding contour plots of the density function $|\phi_g(\mathbf{x})|^2$ of Table 1.

Example 6.8. The choice of the initial data also affects the final converged stationary states. Since all the algorithms we discussed are local minimization algorithms, inappropriate initial guess might lead to local minimum. To illustrate this claim, we take $\varepsilon = 10^{-14}$, $V(\mathbf{x}) = \frac{|\mathbf{x}|^2}{2}$, $\eta = 500$ and compute the ground states of the rotating GPE for different ω and 10 types of frequently used initial data

$$(a)\ \phi_a(\mathbf{x}) = \frac{1}{\sqrt{\pi}} e^{-(x^2+y^2)/2}, \qquad (b)\ \phi_b(\mathbf{x}) = (x+iy)\phi_a(\mathbf{x}), \qquad (\bar{b})\ \phi_{\bar{b}}(\mathbf{x}) = \bar{\phi}_b(\mathbf{x}), \tag{6.5}$$

$$(c)\ \phi_c = \frac{(\phi_a(\mathbf{x}) + \phi_b(\mathbf{x}))/2}{\|(\phi_a(\mathbf{x}) + \phi_b(\mathbf{x}))/2\|}, \qquad (\bar{c})\ \phi_{\bar{c}}(\mathbf{x}) = \bar{\phi}_c(\mathbf{x}), \tag{6.6}$$

$$(d)\ \phi_d = \frac{(1-\omega)\phi_a(\mathbf{x}) + \omega\phi_b(\mathbf{x})}{\|(1-\omega)\phi_a(\mathbf{x}) + \omega\phi_b(\mathbf{x})\|}, \qquad (\bar{d})\ \phi_{\bar{d}}(\mathbf{x}) = \bar{\phi}_d(\mathbf{x}), \tag{6.7}$$

$$(e)\ \phi_e = \frac{\omega\phi_a(\mathbf{x}) + (1-\omega)\phi_b(\mathbf{x})}{\omega\phi_a(\mathbf{x}) + (1-\omega)\phi_b(\mathbf{x})\|}, \qquad (\bar{e})\ \phi_{\bar{e}}(\mathbf{x}) = \bar{\phi}_e(\mathbf{x}), \tag{6.8}$$

$$(f)\ \text{Thomas Fermi approximation (6.3).} \tag{6.9}$$

Another approach to prepare some initial data is as follows: we first consider one of the above initial guess (a)–(f), next compute the ground state on a coarse spatial grid, say with a number of grid points $M_p \times M_p$ (with $M_p = 2^p$), and then denote the corresponding stationary state by ϕ_g^p. We next subdivide the grid with $M_{p+1} \times M_{p+1}$ points, interpolate ϕ_g^p on the refined grid $M_{p+1} \times M_{p+1}$ to get a new initial data and launch the algorithm at level $p+1$, and so on until the finest

Table 2
Example 6.8. Fixed grid approach (with $M = 2^9$): converged energies and the CPU times (*seconds*) for the solution with lowest energy (which is underlined).

ω	(a)	(b)	(b2)	(c)	(c2)	(d)	(d2)	(e)	(e2)	(f)	CPU
0.5	8.5118	8.2606	9.2606	8.0246	8.0197	8.0246	$\underline{8.0197}$	8.0246	8.0197	8.0246	176.0
0.6	8.5118	8.1606	9.3606	7.5845	7.5910	7.5845	$\underline{7.5845}$	7.5845	7.5910	7.5845	310.7
0.7	8.5118	8.0606	9.4606	6.9754	$\underline{6.9731}$	6.9792	6.9754	6.9754	6.9792	6.9767	542.4
0.8	8.5118	7.9606	9.5606	6.1016	$\underline{6.0997}$	6.1031	6.1031	6.1040	6.1019	6.1016	417.0
0.9	8.5118	7.8606	9.6606	4.7777	4.7777	4.7777	$\underline{4.7777}$	4.7777	4.7777	4.7777	1051.1
0.95	8.5118	7.8106	9.7106	3.7414	3.7414	$\underline{3.7414}$	3.7414	3.7414	3.7414	3.7414	3280.5

Table 3
Example 6.8. Multigrid approach (starting from the coarsest level $p = 6$ to the finest level $p = 9$): converged energies and the CPU times (*seconds*) for the solution with lowest energy (underline).

ω	(a)	(b)	(b2)	(c)	(c2)	(d)	(d2)	(e)	(e2)	(f)	CPU
0.5	8.0246	8.0197	8.0197	8.0197	8.0197	8.0197	$\underline{8.0197}$	8.0197	8.0197	8.0257	29.5
0.6	7.5845	7.5845	7.5910	7.5845	7.5910	7.5890	$\underline{7.5845}$	7.5845	7.5910	7.5845	32.3
0.7	6.9767	$\underline{6.9726}$	6.9792	6.9754	6.9731	6.9731	6.9731	6.9757	6.9731	6.9731	53.3
0.8	6.1019	6.1031	6.1019	$\underline{6.0997}$	6.1016	6.1016	6.1016	6.1019	6.1016	6.0997	75.2
0.9	4.7777	4.7777	4.7777	4.7777	4.7777	$\underline{4.7777}$	4.7777	4.7777	4.7777	4.7777	238.1
0.95	3.7414	3.7414	3.7414	3.7414	3.7414	$\underline{3.7414}$	3.7414	3.7414	3.7414	3.7414	621.9

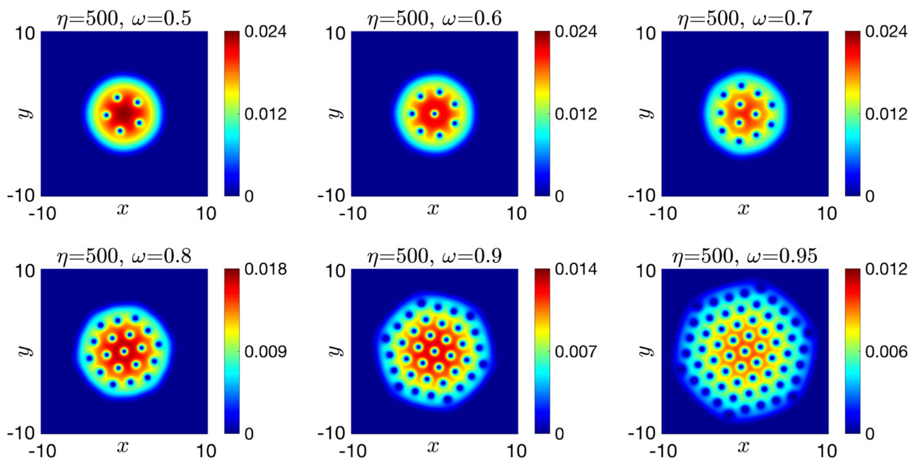

Fig. 10. Contour plots of $|\phi_g(\mathbf{x})|^2$ corresponding to the lowest energy levels in Table 3.

grid with $M \times M$ points where the converged solution is still denoted by ϕ_g. Similar to [47], this multigrid technique is applied here begins with the coarsest grid $M_6 = 2^6$ and ends with the finest grid $M = 2^9$. We use the tolerance parameters $\varepsilon = 10^{-14}$ for $M = 2^9$, and $\varepsilon = 10^{-12}$ for $p = 6, 7, 8$.

Tables 2 and 3 list the energies obtained by PCG$_C$ *via* the fixed and multigrid approaches, respectively, for different initial data and ω. The stationary states $\phi_g(\mathbf{x})$ with lowest energies are marked by underlines and the corresponding CPU times are listed in the same Table. Moreover, Fig. 10 shows the contour plots $|\phi_g(\mathbf{x})|^2$ of the converged solution with lowest energy obtained by the multigrid approach. Now, let us denote by $E_n^p := E(\phi_n^p)$ the evaluated energy at step n for a discretization level $p = 6, 7, 8$, and let $E_g = E(\phi_g)$ the energy for the converged stationary state for the finest grid. Then, we represent on Fig. 11 the evolution of $\log_{10}(|E_n^p - E_g|)$ vs. the CPU time for a rotating velocity $\omega = 0.95$. For comparison, we also show the corresponding evolution obtained by the fixed grid approach. The contour plots of $|\phi_g^p(\mathbf{x})|^2$ obtained for each intermediate coarse grid for $p = 6, 7, 8$, and the initial guess are also reported.

From these tables and figures, we can see that: (i) Usually, the PCG$_C$ algorithm with an initial data of type (d) or $(\bar{d})$ converges to the stationary state of lowest energy; (ii) The multigrid approach is more robust than the fixed grid approach in terms of CPU time and possibility to obtain a stationary state with lower energy.

6.3. Numerical results in 3D

Example 6.9. Here, we apply the PCG$_C$ algorithm to compute some realistic 3D challenging problems. To this end, $V(\mathbf{x})$ is chosen as the harmonic plus quartic potential (6.2), with $\gamma_x = \gamma_y = 1$, $\gamma_z = 3$, $\alpha = 1.4$ and $\kappa = 0.3$. The computational domain is $\mathcal{D} = [-8, 8]^3$ and the mesh size is: $h = \frac{1}{8}$. We test four cases: (i) $\eta = 100$, $\omega = 1.4$; (ii) $\eta = 100$, $\omega = 1.8$; (iii) $\eta = 5000$ and $\omega = 3$; (iv) $\eta = 10000$ and $\omega = 3$. The initial guess is always taken as the Thomas–Fermi initial data and

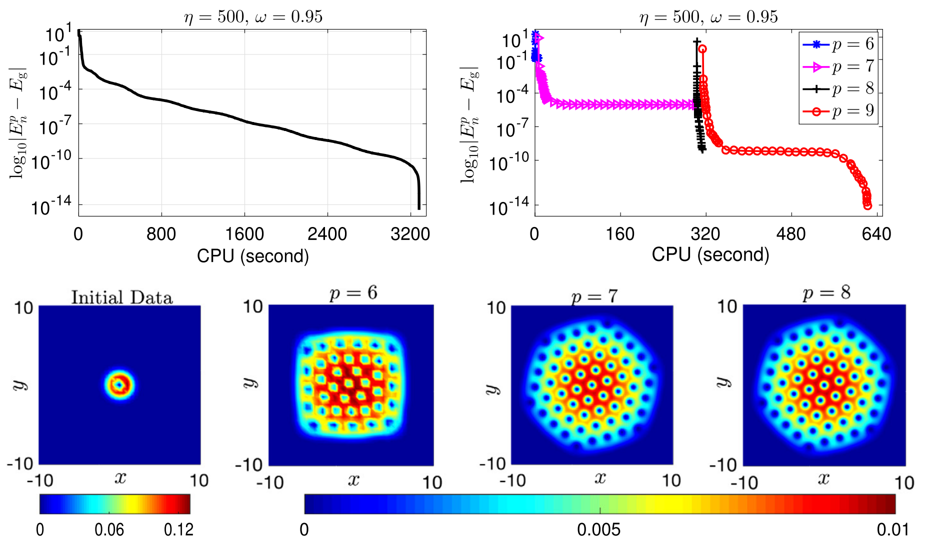

Fig. 11. Example 6.8. Energy error $\log_{10}(|E_n^p - E_g|)$ vs. the accumulated CPU time for $\omega = 0.95$ with initial data (d) in Table 2 ($p = 9$, upper left) and respectively Table 3 (upper right) as well as the stationary state obtained at each intermediate level (lower, $p = 6, 7, 8$).

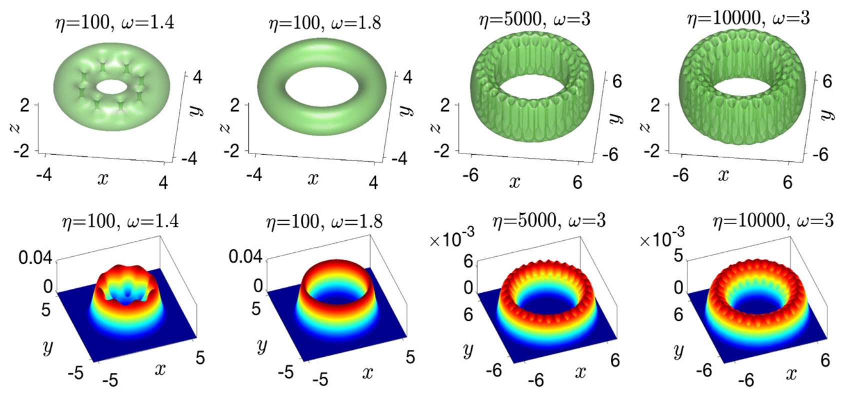

Fig. 12. Example 6.9. Isosurface $|\phi_g(\mathbf{x})|^2 = 10^{-3}$ (upper) and surface plot of $|\phi_g(x, y, z = 0)|^2$ (lower) in Example 6.9. The CPU cost for these four cases are respectively 2256 (s), 1403 (s), 11694 (s) and 21971 (s).

the multigrid algorithm is used. Fig. 12 shows the isosurface $|\phi_g(\mathbf{x})|^2 = 10^{-3}$ and the surface plot of $|\phi_g(x, y, z = 0)|^2$ for the four cases. The CPU times for these four cases are respectively 2256 s, 1403 s, 11694 s and 21971 s.

7. Conclusion

We have introduced a new preconditioned nonlinear conjugate gradient algorithm to compute the stationary states of the GPE with fast rotation and large nonlinearities that arise in the modeling of Bose–Einstein Condensates. The method, which is simple to implement, appears robust and accurate. In addition, it is far more efficient than standard approaches as shown through numerical examples in 1D, 2D and 3D. Furthermore, a simple multigrid approach can still accelerates the performance of the method and leads to a gain of robustness thanks to the initial data. The extension to much more general systems of GPEs is direct and offers an interesting tool for solving highly nonlinear 3D GPEs, even for very large rotations.

Acknowledgements

X. Antoine and Q. Tang thanks the support of the French ANR grant ANR-12-MONU-0007-02 BECASIM (“Modèles Numériques” call).

References

[1] J.R. Abo-Shaeer, C. Raman, J.M. Vogels, W. Ketterle, Observation of vortex lattices in Bose–Einstein condensates, Science 292 (5516) (2001) 476–479.
[2] P.-A. Absil, R. Mahony, R. Sepulchre, Optimization Algorithms on Matrix Manifolds, Princeton University Press, 2009.
[3] S.K. Adhikari, Numerical solution of the two-dimensional Gross–Pitaevskii equation for trapped interacting atoms, Phys. Lett. A 265 (1–2) (2000) 91–96.

[4] M.H. Anderson, J.R. Ensher, M.R. Matthews, C.E. Wieman, E.A. Cornell, Observation of Bose–Einstein condensation in a dilute atomic vapor, Science 269 (5221) (1995) 198–201.
[5] X. Antoine, W. Bao, C. Besse, Computational methods for the dynamics of the nonlinear Schrödinger/Gross–Pitaevskii equations, Comput. Phys. Commun. 184 (12) (2013) 2621–2633.
[6] X. Antoine, R. Duboscq, GPELab, a Matlab toolbox to solve Gross–Pitaevskii equations I: computation of stationary solutions, Comput. Phys. Commun. 185 (11) (2014) 2969–2991.
[7] X. Antoine, R. Duboscq, Robust and efficient preconditioned Krylov spectral solvers for computing the ground states of fast rotating and strongly interacting Bose–Einstein condensates, J. Comput. Phys. 258 (2014) 509–523.
[8] X. Antoine, R. Duboscq, GPELab, a Matlab toolbox to solve Gross–Pitaevskii equations II: dynamics and stochastic simulations, Comput. Phys. Commun. 193 (2015) 95–117.
[9] X. Antoine, R. Duboscq, Modeling and computation of Bose–Einstein condensates: stationary states, nucleation, dynamics, stochasticity, in: C. Besse, J.C. Garreau (Eds.), Nonlinear Optical and Atomic Systems: at the Interface of Physics and Mathematics, in: Lect. Notes Math., vol. 2146, 2015, pp. 49–145.
[10] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, H. van der Vorst, Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide, SIAM, 2000.
[11] W. Bao, Ground states and dynamics of multi-component Bose–Einstein condensates, SIAM J. Multiscale Model. Simul. 2 (2) (2004) 210–236.
[12] W. Bao, Y. Cai, Ground states of two-component Bose–Einstein condensates with an internal atomic Josephson junction, East Asian J. Appl. Math. 1 (2011) 49–81.
[13] W. Bao, Y. Cai, Mathematical theory and numerical methods for Bose–Einstein condensation, Kinet. Relat. Models 6 (1) (2013) 1–135.
[14] W. Bao, Y. Cai, H. Wang, Efficient numerical methods for computing ground states and dynamics of dipolar Bose–Einstein condensates, J. Comput. Phys. 229 (20) (2010) 7874–7892.
[15] W. Bao, Q. Du, Computing the ground state solution of Bose–Einstein condensates by a normalized gradient flow, SIAM J. Sci. Comput. 25 (5) (2004) 1674–1697.
[16] W. Bao, S. Jiang, Q. Tang, Y. Zhang, Computing the ground state and dynamics of the nonlinear Schrödinger equation with nonlocal interactions via the nonuniform FFT, J. Comput. Phys. 296 (2015) 72–89.
[17] W. Bao, W. Tang, Ground-state solution of Bose–Einstein condensate by directly minimizing the energy functional, J. Comput. Phys. 187 (1) (2003) 230–254.
[18] R. Barrett, M. Berry, T. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, H. Van der Vorst, Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, SIAM, 1994.
[19] D. Baye, J.M. Sparenberg, Resolution of the Gross–Pitaevskii equation with the imaginary-time method on a Lagrange mesh, Phys. Rev. E 82 (5) (2010).
[20] C.C. Bradley, C.A. Sackett, J.J. Tollett, R.G. Hulet, Evidence of Bose–Einstein condensation in an atomic gas with attractive interactions, Phys. Rev. Lett. 75 (9) (1995) 1687–1690.
[21] V. Bretin, S. Stock, Y. Seurin, J. Dalibard, Fast rotation of a Bose–Einstein condensate, Phys. Rev. Lett. 92 (5) (2004).
[22] T. Byrnes, K. Wen, Y. Yamamoto, Macroscopic quantum computation using Bose–Einstein condensates, Phys. Rev. A 85 (4) (2012).
[23] M. Caliari, A. Ostermann, S. Rainer, M. Thalhammer, A minimisation approach for computing the ground state of Gross–Pitaevskii systems, J. Comput. Phys. 228 (2) (2009) 349–360.
[24] E. Cances, M. Defranceschi, W. Kutzelnigg, C. Le Bris, Y. Maday, Computational quantum chemistry: a primer, Handb. Numer. Anal. 10 (2003) 3–270.
[25] M.M. Cerimele, M.L. Chiofalo, F. Pistella, S. Succi, M.P. Tosi, Numerical solution of the Gross–Pitaevskii equation using an explicit finite-difference scheme: an application to trapped Bose–Einstein condensates, Phys. Rev. E 62 (1) (2000) 1382–1389.
[26] M.L. Chiofalo, S. Succi, M.P. Tosi, Ground state of trapped interacting Bose–Einstein condensates by an explicit imaginary-time algorithm, Phys. Rev. E 62 (5) (2000) 7438–7444.
[27] R. Courant, K. Friedrichs, H. Lewy, On the partial difference equations of mathematical physics, IBM J. Res. Dev. 11 (2) (1967) 215–234.
[28] F. Dalfovo, S. Giorgini, L.P. Pitaevskii, S. Stringari, Theory of Bose–Einstein condensation in trapped gases, Rev. Mod. Phys. 71 (3) (1999) 463–512.
[29] I. Danaila, F. Hecht, A finite element method with mesh adaptivity for computing vortex states in fast-rotating Bose–Einstein condensates, J. Comput. Phys. 229 (19) (2010) 6946–6960.
[30] I. Danaila, P. Kazemi, A new Sobolev gradient method for direct minimization of the Gross–Pitaevskii energy with rotation, SIAM J. Sci. Comput. 32 (5) (2010) 2447–2467.
[31] K.B. David, M.O. Mewes, M.R. Andrews, N.J. Vandruten, D.S. Durfee, D.M. Kurn, W. Ketterle, Bose–Einstein condensation in gas of sodium atoms, Phys. Rev. Lett. 75 (22) (1995) 3969–3973.
[32] C.M. Dion, E. Cances, Ground state of the time-independent Gross–Pitaevskii equation, Comput. Phys. Commun. 177 (10) (2007) 787–798.
[33] A. Edelman, T.A. Arias, S.T. Smith, The geometry of algorithms with orthogonality constraints, SIAM J. Matrix Anal. Appl. 20 (2) (1998) 303–353.
[34] A.L. Fetter, B. Jackson, S. Stringari, Rapid rotation of a Bose–Einstein condensate in a harmonic plus quartic trap, Phys. Rev. A 71 (2005) 013605.
[35] B.-W. Jeng, Y.-S. Wang, C.-S. Chien, A two-parameter continuation algorithm for vortex pinning in rotating Bose–Einstein condensates, Comput. Phys. Commun. 184 (3) (2013) 493–508.
[36] A.V. Knyazev, Toward the optimal preconditioned eigensolver: locally optimal block preconditioned conjugate gradient method, SIAM J. Sci. Comput. 23 (2) (2001) 517–541.
[37] K.W. Madison, F. Chevy, V. Bretin, J. Dalibard, Stationary states of a rotating Bose–Einstein condensate: routes to vortex nucleation, Phys. Rev. Lett. 86 (20) (2001) 4443–4446.
[38] K.W. Madison, F. Chevy, W. Wohlleben, J. Dalibard, Vortex formation in a stirred Bose–Einstein condensate, Phys. Rev. Lett. 84 (5) (2000) 806–809.
[39] M.R. Matthews, B.P. Anderson, P.C. Haljan, D.S. Hall, C.E. Wieman, E.A. Cornell, Vortices in a Bose–Einstein condensate, Phys. Rev. Lett. 83 (13) (1999) 2498–2501.
[40] M.C. Payne, M.P. Teter, D.C. Allan, T.A. Arias, J.D. Joannopoulos, Iterative minimization techniques for ab initio total-energy calculations: molecular dynamics and conjugate gradients, Rev. Mod. Phys. 64 (4) (1992) 1045–1097.
[41] C. Raman, J.R. Abo-Shaeer, J.M. Vogels, K. Xu, W. Ketterle, Vortex nucleation in a stirred Bose–Einstein condensate, Phys. Rev. Lett. 87 (21) (2001).
[42] Y. Saad, Iterative Methods for Sparse Linear Systems, 2nd edition, SIAM, 2003.
[43] Y. Saad, Numerical Methods for Large Eigenvalue Problems, SIAM, 2011.
[44] Y. Saad, J.R. Chelikowsky, S.M. Shontz, Numerical methods for electronic structure calculations of materials, SIAM Rev. 52 (1) (2010) 3–54.
[45] M.P. Teter, M.C. Payne, D.C. Allan, Solution of Schrödinger's equation for large systems, Phys. Rev. B 40 (1989) 12255–12263.
[46] Y.-S. Wang, B.-W. Jeng, C.-S. Chien, A two-parameter continuation method for rotating two-component Bose–Einstein condensates in optical lattices, Commun. Comput. Phys. 13 (2013) 442–460.
[47] X. Wu, Z. Wen, W. Bao, A regularized Newton method for computing ground states of Bose–Einstein condensates, arXiv:1504.02891, 2015.
[48] C. Yuce, Z. Oztas, Off-axis vortex in a rotating dipolar Bose–Einstein condensate, J. Phys. B, At. Mol. Opt. Phys. 43 (13) (2010).
[49] R. Zeng, Y. Zhang, Efficiently computing vortex lattices in rapid rotating Bose–Einstein condensates, Comput. Phys. Commun. 180 (6) (2009) 854–860.
[50] Y. Zhou, J.R. Chelikowsky, X. Gao, A. Zhou, On the preconditioning function used in planewave DFT calculations and its generalization, Commun. Comput. Phys. 18 (1) (2015) 167–179.