Available online at www.sciencedirect.com
ScienceDirect
Computer Physics Communications 00 (2014) 1–25
Computer Physics Communications

GPELab, a Matlab Toolbox to solve Gross-Pitaevskii Equations II: dynamics and stochastic simulations

Xavier Antoine[a,b], Romain Duboscq[c]

[a]*Université de Lorraine, Institut Elie Cartan de Lorraine, UMR 7502, Vandoeuvre-lès-Nancy, F-54506, France*
[b]*Inria Nancy Grand-Est/IECL-CORIDA, France*
[c]*Institut de Mathématiques de Toulouse, UMR 5219, Université Paul Sabatier Toulouse 3, 118 Route de Narbonne, 31062 Toulouse Cedex 9, France*

Abstract

GPELab is a free Matlab toolbox for modeling and numerically solving large classes of systems of Gross-Pitaevskii equations that arise in the physics of Bose-Einstein condensates. The aim of this second paper, which follows [8], is to first present the various pseudospectral schemes available in GPELab for computing the deterministic and stochastic nonlinear dynamics of Gross-Pitaevskii equations [7]. Next, the corresponding GPELab functions are explained in detail. Finally, some numerical examples are provided to show how the code works for the complex dynamics of BEC problems.

Keywords: Bose-Einstein condensates, nonlinear Schrödinger equation, Gross-Pitaevskii equation, nonlinear dynamics, stochastic dynamics, computational schemes, numerical simulation
PACS: 02.60.-x, 02.70.-c, 31.15.-p, 31.15.xf
2010 MSC: 35Q41, 35R60, 81Q05, 65M12, 65M70, 65Z05

Contents

Email addresses: xavier.antoine@univ-lorraine.fr (Xavier Antoine), Romain.Duboscq@math.univ-toulouse.fr (Romain Duboscq)

0. Program Summary

Manuscript title: GPELab, a Matlab Toolbox to solve Gross-Pitaevskii Equations II: dynamics and stochastic simulations
Authors: Xavier ANTOINE & Romain DUBOSCQ
Program title: GPELab
Licensing provisions: Standard CPC licence
Programming language: Matlab
Computer(s) for which the program has been designed: PC, Mac
Operating system(s) for which the program has been designed: Windows, Mac OS, Linux
RAM required to execute with typical data: 4000 Megabytes
Has the code been vectorised or parallelized?: Yes
Number of processors used: Most if not all
Keywords: Matlab, Bose-Einstein condensates, Gross-Pitaevskii Equation, Dynamical solution, Splitting schemes, Relaxation scheme
CPC Library Classification: 2.7, 4.6, 7.7
Nature of problem: Simulation of dynamical solutions for a class of systems (multi-components) of time-dependent Gross-Pitaevskii equations in 1d, 2d and 3d. This program is particularly well designed for the simulation of the dynamics of Bose-Einstein condensates as well as the computation of ground states.
Solution method: We use spectral schemes in space and splitting/relaxation scheme in time.
Running time: From a few minutes for simple problems to few hours for more complex situations on a medium computer.

1. Introduction

`GPELab`[1] (`Gross-Pitaevskii Equation Laboratory`) is an open access Matlab toolbox [8] developed for computing *the stationary states and the nonlinear (deterministic and stochastic) dynamics of 1d-2d-3d Gross-Pitaevskii Equations* (GPEs) [24, 32, 33, 40, 41]. The GPE is widely used for modeling Bose-Einstein Condensates (BECs) [6, 11, 17, 18]. GPELab can treat complex physics problems including general potentials [25, 26, 29, 48], local and nonlocal (dipole-dipole) nonlinearities [21, 22, 23, 30, 39], rotation effects [4, 34, 35, 36, 42], stochastic terms [1, 2, 3, 19,

[1] `http://gpelab.math.cnrs.fr/`

20, 44] and/or multi-components problems [5, 27, 31, 37, 38, 43, 49]. The idea behind GPELab is to propose to physicists working on BECs a simple, generic and robust computational tool for modeling a wide class of Bose-Einstein condensates. GPELab uses pseudospectral approximation techniques [9, 11, 12, 50] which provide highly accurate spatial solutions compared e.g. with finite difference schemes. In the first paper [8], we introduced the numerical methods for computing the stationary states of GPEs and the most important functions that are defined in GPELab. The aim of this second paper is to describe the numerical schemes that are included in GPELab for solving the nonlinear deterministic and stochastic dynamics of Gross-Pitaevskii equations [1, 2, 3, 9, 19], the associated GPELab functions and to give in detail some numerical examples.

The paper is organized as follows. After the introduction of the dimensionless GPE (section 2), we describe in section 3 the numerical spectral schemes (Time Splitting and relaxation schemes) that are used in GPELab for the discretization of deterministic or stochastic systems of GPEs. The associated functions are described in section 4. Three numerical examples are fully developed in section 5. Finally, section 6 concludes.

2. The dimensionless Gross-Pitaevskii equation used in GPELab

By using some suitable changes of variables [8, 11], it can be proved that the GPE coming from physics can be rewritten as a dimensionless GPE in dimension d following

$$i\frac{\partial\psi}{\partial t} = \left(-\frac{1}{2}\Delta + V_d + \beta_d|\psi|^2 - \Omega L_z\right)\psi. \tag{2.1}$$

The unknown $\psi(t,\mathbf{x})$ is the condensate wave function that depends on the time $t > 0$ and spatial variable $\mathbf{x} \in \mathbb{R}^d$. The operator Δ is the standard laplacian operator, $V_d(= V)$ is the potential function which is $(\mathbf{x}, t)$-dependent and the nonlinearity strength is $\beta_d(= \beta)$. The positive real-valued parameter Ω is the rotation speed and $L_z = -i(x\partial_y - y\partial_x)$ is the rotating operator in the two- and three-dimensional cases. The dimensionless energy functional $E_{\beta,\Omega}$ is defined [8, 13] by

$$E_{\beta,\Omega}(\psi) = \int_{\mathbb{R}^d}\left[\frac{1}{2}|\nabla\psi|^2 + V|\psi|^2 + \frac{\beta}{2}|\psi|^4 - \Omega\psi^* L_z\psi\right]d\mathbf{x}, \tag{2.2}$$

where ψ^* is the complex conjugate function of ψ. We refer to [8] for more details about the notations, examples of potentials and nonlinearities that can be defined in GPELab.

3. Spectral schemes for the simulation of the dynamics

3.1. Alternate Direction Implicit-Time Splitting pseudo SPectral (ADI-TSSP) schemes

Let us introduce A and B, two self-adjoint operators such that: $\mathcal{D}(A) \subset L^2$, $\mathcal{D}(B) \subset L^2$ and $A + B$ is a self-adjoint operator on $\mathcal{D}(A) \cap \mathcal{D}(B)$. We designate by $\mathcal{D}(A)$ and $\mathcal{D}(B)$ the domains of the operators A and B, respectively. We consider the following time-dependent Partial Differential Equation (PDE)

$$\begin{cases} \partial_t\psi(t,\mathbf{x}) = A\psi(t,\mathbf{x}) + B\psi(t,\mathbf{x}), \\ \psi(0,\mathbf{x}) = \psi_0(\mathbf{x}), \end{cases}$$

and denote by $\psi(t,\mathbf{x}) = e^{(A+B)t}\psi_0(\mathbf{x})$ its solution, for all $t > 0$ and $\mathbf{x} \in \mathbb{R}^d$. The Time-Splitting (TS) schemes consist in approximating the solution ψ *via* a splitting of the exponential operator $e^{(A+B)t}$ involving the operators e^{At} and e^{Bt}. Let us write

$$\psi(t+\delta t,\mathbf{x}) = e^{(A+B)\delta t}\psi(t,\mathbf{x}) \approx e^{a_1 A\delta t}e^{b_1 B\delta t}e^{a_2 A\delta t}e^{b_2 B\delta t}...e^{a_p A\delta t}e^{b_p B\delta t}\psi(t,\mathbf{x}),$$

where $\{a_k, b_k\}_{1\le k\le p} \subset \mathbb{R}$ are weights that are computed to get an approximation of $e^{(A+B)\delta t}$ of a given order for a time step $\delta t \ll 1$ and $t \in \mathbb{R}^+ := \{t > 0\}$. The most commonly used time-splitting schemes are the Lie ($a_1 = b_1 = 1$) (see subsection 3.1.1) and the Strang ($a_1 = a_2 = 1/2$ and $b_1 = 1, b_2 = 0$) (see subsection 3.1.2) schemes. They are respectively of order one and two. Higher-order schemes [47] can be constructed with appropriately chosen weights

$\{a_k, b_k\}_{1\le k\le p}$. The motivation behind the splitting schemes lies in the fact that the equations associated to the operators A and B can be efficiently solved. In GPELab, the standard decomposition [14, 15]

$$A = \frac{i}{2}\Delta + i\Omega L_z, \qquad B = -iV(t,\mathbf{x}) - i\beta|\psi(t,\mathbf{x})|^2,$$

is used. As seen later, the PDE associated to A can be solved by using an *Alternate Direction Implicit* method and Fast Fourier Transforms (FFTs) [15]. In addition, the ODE related to the nonlinearity and the potential parts can be integrated exactly.

3.1.1. The Lie ADI-TSSP scheme

Lie scheme. The Lie splitting scheme uses the following approximation

$$\psi(t+\delta t,\mathbf{x}) \approx e^{i(1/2\Delta+\Omega L_z)\delta t} e^{-i(V(t,\mathbf{x})+\beta|\psi|^2)\delta t}\psi(t,\mathbf{x}). \tag{3.3}$$

For an initial data $\psi^0 = \psi(0,\mathbf{x}) \in L^2(\mathbb{R}^d)$, we want to numerically solve (2.1) on the time interval $[0,T]$ ($T>0$) which is uniformly discretized. We assume that: $N\delta t = T$, with $N \in \mathbb{N}$ and $\delta t > 0$ and $N \in \mathbb{N}$. For $0 \le n \le N$ and $t_n = n\delta t$, the Lie scheme leads to the two-steps method

1) First, compute the solution ψ_1 to the PDE

$$\begin{cases} i\partial_t\psi_1(t,\mathbf{x}) = -\dfrac{1}{2}\Delta\psi_1(t,\mathbf{x}) - \Omega L_z\psi_1(t,\mathbf{x}),\ t_n < t \le t_{n+1},\ \forall \mathbf{x}\in\mathbb{R}^d, \\ \psi_1(t_n,\cdot) = \psi^n. \end{cases} \tag{3.4}$$

2) Next, determine ψ_2 solution to the ODE

$$\begin{cases} i\partial_t\psi_2(t,\mathbf{x}) = V(t,\mathbf{x})\psi_2(t,\mathbf{x}) + \beta|\psi_2(t,\mathbf{x})|^2\psi_2(t,\mathbf{x}),\ t_n < t \le t_{n+1},\ \forall \mathbf{x}\in\mathbb{R}^d, \\ \psi_2(t_n,\cdot) = \psi_1(t_{n+1},\cdot). \end{cases} \tag{3.5}$$

Finally, we set: $\psi^{n+1} := \psi_2(t_{n+1},\cdot)$, which is an approximation of $\psi(t_{n+1},\cdot)$.

Time discretization and the ADI technique. To simplify the presentation, we consider the two-dimensional case ($\mathbf{x} = (x,y) \in \mathbb{R}^2$) (but the 1d and 3d cases can be easily deduced). For a non rotating BEC ($\Omega = 0$), Eq. (3.4) can be efficiently solved by inverting the Laplacian operator through FFTs. For $\Omega \neq 0$, we cannot proceed in the same way since the operator $L_z = i(y\partial_x - x\partial_y)$ is not diagonal in the Fourier space. In [15], Bao *et al.* propose to use an ADI method to avoid this problem. This basic idea consists in splitting the derivative operators with respect to each direction into successive equations. This allows the use of one-dimensional FFTs for solving each equation. Finally, equation (3.4) is decomposed into the two following steps

1.a) first, find $\psi^{(1)}$ such that

$$\begin{cases} i\partial_t\psi^{(1)}(t,\mathbf{x}) = -\dfrac{1}{2}\partial_{xx}\psi^{(1)}(t,\mathbf{x}) - i\Omega y\partial_x\psi^{(1)}(t,\mathbf{x}),\ t_n < t \le t_{n+1},\ \forall \mathbf{x}\in\mathbb{R}^2, \\ \psi^{(1)}(t_n,\cdot) = \psi^n(\cdot), \end{cases} \tag{3.6}$$

1.b) and then, compute $\psi^{(2)}$ as the solution to the equation

$$\begin{cases} i\partial_t\psi^{(2)}(t,\mathbf{x}) = -\dfrac{1}{2}\partial_{yy}\psi^{(2)}(t,\mathbf{x}) + i\Omega x\partial_y\psi^{(2)}(t,\mathbf{x}),\ t_n < t \le t_{n+1},\ \forall \mathbf{x}\in\mathbb{R}^2, \\ \psi^{(2)}(t_n,\cdot) = \psi^{(1)}(t_{n+1},\cdot). \end{cases} \tag{3.7}$$

Next, Eq. (3.5) requires to solving the following ODE

$$\begin{cases} i\partial_t\psi^{(3)}(t,\mathbf{x}) = V(t,\mathbf{x})\psi^{(3)}(t,\mathbf{x}) + \beta|\psi^{(3)}(t,\mathbf{x})|^2\psi^{(3)}(t,\mathbf{x}),\ t_n < t \le t_{n+1},\ \forall \mathbf{x}\in\mathbb{R}^2, \\ \psi^{(3)}(t_n,\mathbf{x}) = \psi^{(2)}(t_{n+1},\mathbf{x}), \end{cases} \tag{3.8}$$

whose solution is

$$\psi^{(3)}(t,\mathbf{x}) = \psi^{(2)}(t_{n+1},\mathbf{x})e^{-i\beta|\psi^{(2)}(t_{n+1},\mathbf{x})|^2(t-t_n)-i\int_{t_n}^{t}V(s,\mathbf{x})ds}, \tag{3.9}$$

which finally gives: $\psi^{n+1}(\mathbf{x}) \approx \psi^{(3)}(t_{n+1},\mathbf{x})$.

The ADI technique implies a loss of symmetry of the scheme when solving the partial differential operators of Eq. (3.4). Indeed, we first integrate in the x-direction in (3.6) and next in the y-direction according to (3.7). To symmetrize the scheme, we alternate the ordering of the derivative directions any two time steps. Concretely, from t_n to t_{n+1}, (3.6) is solved and next equation (3.7) followed by (3.8). From t_{n+1} to t_{n+2}, (3.7) is first solved, then equation (3.6) and finally again Eq. (3.8).

Space discretization in 2d and implementation. GPELab considers an approach based on Fourier series representations through FFTs [8]. Periodic boundary conditions are set on the fictitious boundary of a large enough finite computational box: $\mathcal{O} :=]-a_x;a_x[\times]-a_y;a_y[$. Let us introduce: $\mathcal{P}_{J,K} = \left\{(j,k)\in\mathbb{N}^2; 0\le j\le J-1 \text{ and } 0\le k\le K-1\right\}$, with $J,K\ge 2$, and two uniform discretization steps h_x and h_y in the x- and y-directions, respectively. The partial Fourier pseudospectral discretizations in the x- and y-directions are respectively given by

$$\psi(t,x_j,y_k) = \frac{1}{J}\sum_{p=-J/2}^{J/2-1}\widehat{\psi_p}(t,y_k)e^{i\mu_p(x_j+a_x)}, \quad \psi(t,x_j,y_k) = \frac{1}{K}\sum_{q=-K/2}^{K/2-1}\widehat{\psi_q}(t,x_j)e^{i\lambda_q(y_k+a_y)}, \tag{3.10}$$

$\forall t\in\mathbb{R}^+$ and $\forall (j,k)\in\mathcal{P}_{J,K}$ and where

$$\widehat{\psi_p}(t,y_k) = \sum_{j=0}^{J-1}\psi(t,x_j,y_k)e^{-i\mu_p(x_j+a_x)}, \quad \widehat{\psi_q}(t,x_j) = \sum_{k=0}^{K-1}\psi(t,x_j,y_k)e^{-i\lambda_q(y_k+a_y)}, \tag{3.11}$$

with $\mu_p = \frac{\pi p}{L_x}$ and $\lambda_q = \frac{\pi q}{L_y}$. By using (3.10) and (3.11), the partial differential operators in the x- and y-directions are discretized as

$$\forall (j,k)\in\mathcal{P}_{J,K}, \quad \partial_x\psi(t,x_j,y_k) \approx \frac{1}{J}\sum_{p=-J/2}^{J/2-1} i\mu_p\widehat{\psi_p}(t,y_k)e^{i\mu_p(x_j+a_x)}, \quad \partial_y\psi(t,x_j,y_k) \approx \frac{1}{K}\sum_{q=-K/2}^{K/2-1} i\lambda_q\widehat{\psi_q}(t,x_j)e^{i\lambda_q(y_k+a_y)}.$$

Thus, $\forall t\in[t_n,t_{n+1}]$, $0\le k\le K-1$ and $1-J/2\le p\le J/2$,we obtain

$$i\partial_t\widehat{\psi}_p^{(1)}(t,y_k) = \left(\frac{1}{2}\mu_p^2 + \Omega y\mu_p\right)\widehat{\psi}_p^{(1)}(t,y_k).$$

This ODE can be exactly integrated in time

$$\forall t\in[t_n,t_{n+1}], \widehat{\psi}_p^{(1)}(t,y_k) = e^{-i\left(\frac{1}{2}\mu_p^2+\Omega y\mu_p\right)(t-t_n)}\widehat{\psi}_p^{(1)}(t_n,y_k).$$

Similarly, for Eq. (3.7), one gets

$$1-K/2\le q\le K/2, \quad \widehat{\psi}_q^{(2)}(t,x_j) = e^{-i\left(\frac{1}{2}\lambda_q^2-\Omega x\lambda_q\right)(t-t_n)}\widehat{\psi}_q^{(2)}(t_n,x_j).$$

Thus, the first steps 1.a)-1.b) for solving Eqs. (3.6)-(3.7) on $[t_n,t_{n+1}]$ and for the spatial grid $(x_j,y_k)_{(j,k)\in\mathcal{P}_{J,K}}$ express as

$$\psi^{(1)}(t_{n+1},x_j,y_k) = \frac{1}{J}\sum_{p=-J/2}^{J/2-1} e^{-i\left(\frac{1}{2}\mu_p^2+\Omega y_k\mu_p\right)\delta t}\,\widehat{\psi_p^n}(y_k)e^{i\mu_p(x_j+L_x)},$$

$$\psi^{(2)}(t_{n+1},x_j,y_k) = \frac{1}{K}\sum_{q=-K/2}^{K/2-1} e^{-i\left(\frac{1}{2}\lambda_q^2-\Omega x_j\lambda_q\right)\delta t}\,\widehat{\psi}_q^{(1)}(t_{n+1},x_j)e^{i\lambda_q(y_k+L_y)}.$$

In GPELab, these operations are based on the `fft()` and `ifft()` Matlab functions. Moreover, the exponential matrix is computed by the usual exponential Matlab function. The discretization of (3.9) uses the standard Simpson's quadrature rule

$$\int_{t_n}^{t_{n+1}} V(s,x_j,y_k)ds \approx \frac{1}{6}\left(V(t_n,x_j,y_k) + 6V\left(t_{n+1/2},x_j,y_k\right) + V(t_{n+1},x_j,y_k)\right)(t_{n+1}-t_n) := \tilde{V}_n(x_j,y_k)\delta t,$$

with $t_{n+1/2} = (t_n + t_{n+1})/2$, leading to

$$\psi^{(3)}(t_{n+1}, x_j, y_k) = \psi^{(2)}(t_{n+1}, x_j, y_k) e^{-i\left(\beta|\psi^{(2)}(t_{n+1}, x_j, y_k)|^2 + \bar{V}_n(x_j, y_k)\right)\delta t}.$$

This corresponds to a phase shift of the solution. Let us also remark that everything extend to a general nonlinearity $f(|\psi|, \mathbf{x})$. This scheme, which is called Lie ADI-TSSP scheme, is globally first-order in time and spectrally accurate in space. The computational cost is $O(M \log M)$, setting $M = JK$.

3.1.2. The Strang ADI-TSSP scheme

We now briefly explain the Strang ADI-TSSP scheme since its derivation is similar to the Lie ADI-TSSP scheme. For a time step δt, the approximation of the solution ψ is either

$$\psi(t + \delta t, \mathbf{x}) \approx e^{-i(V(t,\mathbf{x}) + \beta|\psi|^2)\delta t/2} e^{i(1/2\Delta + \Omega L_z)\delta t} e^{-i(V(t,\mathbf{x}) + \beta|\psi|^2)\delta t/2} \psi(t, \mathbf{x}),$$

or

$$\psi(t + \delta t, \mathbf{x}) \approx e^{i(1/2\Delta + \Omega L_z)\delta t/2} e^{-i(V(t,\mathbf{x}) + \beta|\psi|^2)\delta t} e^{i(1/2\Delta + \Omega L_z)\delta t/2} \psi(t, \mathbf{x}).$$

Here, we consider the second formulation. Indeed, in this case, the symmetrization of the method can be directly done on a single step by changing the direction of the third exponential operator to the first one while this is not possible for the second formulation. The resulting Strang ADI-TSSP scheme is given by the successive operations

1) Solve the equation

$$\begin{cases} i\partial_t \psi^{(1)}(t, \mathbf{x}) = -\dfrac{1}{2}\partial_{xx}\psi^{(1)}(t, \mathbf{x}) - i\Omega y \partial_x \psi^{(1)}(t, \mathbf{x}), \ t_n < t \le t_{n+1/2}, \ \forall \mathbf{x} \in \mathbb{R}^2, \\ \psi^{(1)}(t_n, \cdot) = \psi_n(\cdot). \end{cases} \tag{3.12}$$

2) Find $\psi^{(2)}$ such that

$$\begin{cases} i\partial_t \psi^{(2)}(t, \mathbf{x}) = -\dfrac{1}{2}\partial_{yy}\psi^{(2)}(t, \mathbf{x}) + i\Omega x \partial_y \psi^{(2)}(t, \mathbf{x}), \ t_n < t \le t_{n+1/2}, \ \forall \mathbf{x} \in \mathbb{R}^2, \\ \psi^{(2)}(t_n, \cdot) = \psi^{(1)}(t_{n+1/2}, \cdot). \end{cases} \tag{3.13}$$

3) Determine $\psi^{(3)}$ which solves

$$\begin{cases} i\partial_t \psi^{(3)}(t, \mathbf{x}) = V(t, \mathbf{x})\psi^{(3)}(t, \mathbf{x}) + \beta|\psi^{(3)}(t, \mathbf{x})|^2 \psi^{(3)}(t, \mathbf{x}), \ t_n < t \le t_{n+1}, \ \forall \mathbf{x} \in \mathbb{R}^2, \\ \psi^{(3)}(t_n, \cdot) = \psi^{(2)}(t_{n+1/2}, \cdot). \end{cases} \tag{3.14}$$

4) Compute the solution $\psi^{(4)}$ of the equation

$$\begin{cases} i\partial_t \psi^{(4)}(t, \mathbf{x}) = -\dfrac{1}{2}\partial_{yy}\psi^{(4)}(t, \mathbf{x}) + i\Omega x \partial_y \psi^{(4)}(t, \mathbf{x}), \ t_{n+1/2} < t \le t_{n+1}, \ \forall \mathbf{x} \in \mathbb{R}^2, \\ \psi^{(4)}(t_{n+1/2}, \cdot) = \psi^{(3)}(t_{n+1}, \cdot). \end{cases} \tag{3.15}$$

5) Finally, obtain the solution $\psi^{(5)}$ of the equation

$$\begin{cases} i\partial_t \psi^{(5)}(t, \mathbf{x}) = -\dfrac{1}{2}\partial_{xx}\psi^{(5)}(t, \mathbf{x}) - i\Omega y \partial_x \psi^{(5)}(t, \mathbf{x}), \ t_{n+1/2} < t \le t_{n+1}, \ \forall \mathbf{x} \in \mathbb{R}^2, \\ \psi^{(5)}(t_{n+1/2}, \mathbf{x}) = \psi^{(4)}(t_{n+1}, \mathbf{x}). \end{cases} \tag{3.16}$$

This last step finally gives $\psi^{n+1}(\mathbf{x}) := \psi^{(5)}(t_{n+1/2}, \mathbf{x})$. Each PDE with respect to x or y is solved through FFTs and iFFTs.

The Strang ADI-TSSP scheme is second-order in time and spectrally accurate in space for a computational cost $O(M \log M)$. The extensions to the 1d-3d cases are straightforward. Some interesting properties are related to the fact that these schemes are time reversible, mass conserving, time transverse invariant and the dispersion relation holds [7]. Unfortunately, the energy is not exactly conserved (for example when $\Omega = 0$) [7]. Finally, the Lie and Strang ADI-TSSP schemes are unconditionally stable.

3.1.3. Extension of the TSSP schemes to the multi-components case

The TSSP schemes can be extended to the multi-components case, i.e. a system of N_c coupled GPEs. For $\mathbf{x} := (x_1, ..., x_d) \in \mathbb{R}^d$ and with $N_c \in \mathbb{N}^*$, we denote by $\Psi = (\psi_1, ..., \psi_{N_c})$ a vector of N_c wave functions solution to the following system of GPEs

$$i\partial_t \Psi(t,\mathbf{x}) = -\frac{1}{2}\Delta\Psi(t,\mathbf{x}) + V(\mathbf{x})\Psi(t,\mathbf{x}) + \sum_{j=1}^{d} G^j(\mathbf{x}\backslash x_j)\partial_{x_j}\Psi(t,\mathbf{x}) + \beta F(\Psi,\mathbf{x})\Psi(t,\mathbf{x}),\ t>0, \mathbf{x}\in\mathbb{R}^d, \tag{3.17}$$

where we set $\mathbf{x}\backslash x_j = (x_1, ..., x_{j-1}, x_{j+1}, ..., x_d)$. Concerning the definition of the operators occurring in the previous equation, we refer to [8]. We furthermore assume that V and F are two symmetric operators to get the mass conservation property

$$N(\Psi) := \sum_{j=1}^{N_c} N(\psi_j) = \sum_{j=1}^{N_c} \int_{\mathbb{R}^d} |\psi_j(t,\mathbf{x})|^2 d\mathbf{x} = \sum_{j=1}^{N_c} \int_{\mathbb{R}^d} |\psi_j(0,\mathbf{x})|^2 d\mathbf{x} = \|\Psi\|_0^2 = 1,$$

that is, we suppose that $V_{\ell,m} = V_{m,\ell}$ and $F_{\ell,m} = F_{m,\ell}$, $1 \le \ell, m \le N_c$. For the TSSP schemes included in GPELab, we assume that F only depends on the amplitude of Ψ, i.e. $F(\Psi,\mathbf{x}) := F(|\Psi|,\mathbf{x})$, where we define $|\Psi| = (\sum_{j=1}^{N_c} |\psi_j|^2)^{1/2}$. Let us note that an important point here is that we suppose that the variable coefficients matrices in front of the gradients have the following expressions

$$G^j(\mathbf{x}\backslash x_j) = \begin{pmatrix} G^j_{11}(\mathbf{x}\backslash x_j) & G^j_{12}(\mathbf{x}\backslash x_j) & \cdots & G^j_{1N_c}(\mathbf{x}\backslash x_j) \\ G^j_{21}(\mathbf{x}\backslash x_j) & G^j_{22}(\mathbf{x}\backslash x_j) & \cdots & G^j_{2N_c}(\mathbf{x}\backslash x_j) \\ \vdots & \vdots & \ddots & \vdots \\ G^j_{N_c1}(\mathbf{x}\backslash x_j) & G^j_{N_c2}(\mathbf{x}\backslash x_j) & \cdots & G^j_{N_cN_c}(\mathbf{x}\backslash x_j) \end{pmatrix}.$$

The initial data is $\Psi(t=0,\mathbf{x}) = \Psi_0(\mathbf{x})$ and therefore: $\Psi^0(\mathbf{x}) := \Psi_0(\mathbf{x})$. If Ψ^n designates the approximation of the solution at t_n, then the Lie TSSP scheme yields the two successive steps

1) First, solve the equation

$$\begin{cases} i\partial_t \Psi^{(1)}(t,\mathbf{x}) = -\frac{1}{2}\Delta\Psi^{(1)}(t,\mathbf{x}) + \sum_{j=1}^{d} G^j(\mathbf{x}\backslash x_j)\partial_{x_j}\Psi^{(1)}(t,\mathbf{x}),\ t_n < t \le t_{n+1}, \forall \mathbf{x}\in\mathbb{R}^d, \\ \Psi^{(1)}(t_n,\cdot) = \Psi^n(\cdot). \end{cases} \tag{3.18}$$

2) Next, determine the solution to

$$\begin{cases} i\partial_t \Psi^{(2)}(t,\mathbf{x}) = V(\mathbf{x})\Psi^{(2)}(t,\mathbf{x}) + \beta F(|\Psi^{(2)}|,\mathbf{x})\Psi^{(2)}(t,\mathbf{x}),\ t_n < t \le t_{n+1}, \forall \mathbf{x}\in\mathbb{R}^d, \\ \Psi^{(2)}(t_n,\cdot) = \Psi^{(1)}(t_{n+1},\cdot). \end{cases} \tag{3.19}$$

Finally, we set: $\Psi^{n+1}(\cdot) := \Psi^{(2)}(t_{n+1},\cdot)$. Thanks to the assumptions on the potential and the nonlinear operators, we remark that the solution to (3.19) has a modulus which is conserved.

Lemma 1. *For every t in $[t_n, t_{n+1}]$, we have: $|\Psi^{(2)}(t,\mathbf{x})| = |\Psi^{(2)}(t_n,\mathbf{x})|$.*

Proof. First, we can write that

$$\sum_{m=1}^{N_c} \partial_t |\Psi^{(2)}_m(t,\mathbf{x})|^2 = 2\sum_{m=1}^{N_c} \Re\left(\Psi^{(2)}_m(t,\mathbf{x})^* \partial_t \Psi^{(2)}_m(t,\mathbf{x})\right) = -2\sum_{m,o=1}^{N_c} \Im\left(\Psi^{(2)}_m(t,\mathbf{x})^* (V_{mo}(\mathbf{x}) + F_{mo}(|\Psi^{(2)}(t,\mathbf{x})|,\mathbf{x}))\Psi^{(2)}_o(t,\mathbf{x})\right).$$

By using $V_{mo}(\mathbf{x}) = V_{om}(\mathbf{x})$ and $F_{mo}(|\Psi^{(2)}(t,\mathbf{x})|,\mathbf{x}) = F_{om}(|\Psi^{(2)}(t,\mathbf{x})|,\mathbf{x})$, we obtain

$$\begin{aligned}\sum_{m=1}^{N_c} \partial_t |\Psi_m^{(2)}(t,\mathbf{x})|^2 = \\ -2 \sum_{N_c \geq o > m \geq 1} \Im\left((V_{mo}(\mathbf{x}) + F_{mo}(|\Psi^{(2)}(t,\mathbf{x})|,\mathbf{x}))(\Psi_m^{(2)}(t,\mathbf{x})^* \Psi_o^{(2)}(t,\mathbf{x}) + \Psi_o^{(2)}(t,\mathbf{x})^* \Psi_m^{(2)}(t,\mathbf{x}))\right) \\ -2 \sum_{N_c \geq m \geq 1} \Im\left((V_{mm}(\mathbf{x}) + F_{mm}(|\Psi^{(2)}(t,\mathbf{x})|,\mathbf{x}))|\Psi_m^{(2)}(t,\mathbf{x})|^2\right) \\ = -4 \sum_{N_c \geq o > m \geq 1} \Im\left((V_{mo}(\mathbf{x}) + F_{mo}(|\Psi^{(2)}(t,\mathbf{x})|,\mathbf{x}))\Re(\Psi_m^{(2)}(t,\mathbf{x})^* \Psi_o^{(2)}(t,\mathbf{x}))\right) = 0.\end{aligned} \tag{3.20}$$

This concludes the proof. □

Let us explicitly write the method in the 2d case. A first step is to solve the following equation

$$\begin{cases} i\partial_t \Psi^{(1)}(t,\mathbf{x}) = -\dfrac{1}{2}\Delta \Psi^{(1)}(t,\mathbf{x}) + \displaystyle\sum_{j=1}^{2} G^j(\mathbf{x}\backslash x_j)\partial_{x_j}\Psi^{(1)}(t,\mathbf{x}),\ t_n < t \leq t_{n+1}, \forall \mathbf{x} \in \mathbb{R}^2, \\ \Psi^{(1)}(t_n,\cdot) = \Psi^n(\cdot). \end{cases} \tag{3.21}$$

As in the one-component case, an ADI method must be used to decouple the effects of the operators $G^j(\mathbf{x}\backslash x_j)\partial_{x_j}$ in each direction. Therefore, we split Eq. (3.21) in two equations and we simply need to compute the solution of the equation

$$\begin{cases} i\partial_t \Psi^{(1,1)}(t,\mathbf{x}) = -\dfrac{1}{2}\partial_{xx}\Psi^{(1,1)}(t,\mathbf{x}) + G^1(y)\partial_x \Psi^{(1,1)}(t,\mathbf{x}),\ t_n < t \leq t_{n+1}, \forall \mathbf{x} \in \mathbb{R}^2, \\ \Psi^{(1,1)}(t_n,\cdot) = \Psi^n(\cdot), \end{cases} \tag{3.22}$$

then determine $\Psi^{(1,2)}$ such that

$$\begin{cases} i\partial_t \Psi^{(1,2)}(t,\mathbf{x}) = -\dfrac{1}{2}\partial_{yy}\Psi^{(1,2)}(t,\mathbf{x}) + G^2(x)\Psi^{(1,2)}(t,\mathbf{x}),\ t_n < t \leq t_{n+1}, \forall \mathbf{x} \in \mathbb{R}^2, \\ \Psi^{(1,2)}(t_n,\cdot) = \Psi^{(1,1)}(t_{n+1},\cdot). \end{cases} \tag{3.23}$$

A second step is to solve Eq. (3.19) for a well-chosen initial data

$$\begin{cases} i\partial_t \Psi^{(2)}(t,\mathbf{x}) = V(\mathbf{x})\Psi^{(2)}(t,\mathbf{x}) + \beta F(|\Psi^{(2)}(t,\mathbf{x})|,\mathbf{x})\Psi^{(2)}(t,\mathbf{x}),\ t_n < t \leq t_{n+1}, \forall \mathbf{x} \in \mathbb{R}^2, \\ \Psi^{(2)}(t_n,\cdot) = \Psi^{(1,2)}(t_{n+1},\cdot). \end{cases} \tag{3.24}$$

Thanks to Lemma 1, the solution is given by

$$\Psi^{(2)}(t,\mathbf{x}) = e^{-i\beta F(|\Psi^{(1,2)}(t_{n+1},\mathbf{x})|,\mathbf{x})(t-t_n) - iV(\mathbf{x})(t-t_n)}\Psi^{(1,2)}(t_{n+1},\mathbf{x}). \tag{3.25}$$

This finally gives the approximation: $\Psi^{n+1}(\mathbf{x}) \approx \Psi^{(2)}(t_{n+1},\mathbf{x})$. Let us remark that we have to compute the exponential of a matrix to effectively evaluate (3.25). For the full approximation, we adapt to each component the spectral approximation based on `fft` and `ifft` in space. For both the Lie and Strang schemes, the spectral approximation is written under a symmetrical form as for the one-component case.

3.2. Relaxation pseudo SPectral scheme (ReSP)

Introduced by Besse in [16], the relaxation scheme is inspired by the Crank-Nicolson scheme but without solving a nonlinear equation (through a fixed point or a Newton-Raphson method). This dramatically reduces the computational cost of the scheme without loosing the second-order accuracy in time, unconditional stability and mass/energy conservation [7].

3.2.1. Relaxation pseudo SPectral scheme (ReSP) for the rotating GPE

The relaxation scheme applied to equation (2.1) is given by

$$\begin{cases} \dfrac{\phi^{n+1/2}+\phi^{n-1/2}}{2} = \beta|\psi^n|^2, \\ i\dfrac{\psi^{n+1}-\psi^n}{\delta t} = (-\dfrac{1}{2}\Delta - \Omega L_z + \phi^{n+1/2})\left(\dfrac{\psi^{n+1}+\psi^n}{2}\right) + \dfrac{V^{n+1}\psi^{n+1}+V^n\psi^n}{2}, \end{cases} \tag{3.26}$$

where $\phi^{n+1/2} = \phi(t_{n+1/2}, \mathbf{x})$, $\psi^n = \psi(t_n, \mathbf{x})$, $V^n = V(t_n, \mathbf{x})$ and the initial conditions are $\psi^0 = \psi_0$ and $\phi^{-1/2} = \beta|\psi_0|^2$. The extension to a general nonlinearity is direct.

We now have to discretize the operator $-\Delta - \Omega L_z$. To this end, we use the pseudospectral approximation of the spatial derivatives based on the Fourier series expansions (3.10)-(3.11) (see [8]). We have the following discretization

$$\begin{cases} [[\boldsymbol{\phi}^{n+1/2}]] = 2\beta[[|\boldsymbol{\psi}^n|^2]] - [[\boldsymbol{\phi}^{n-1/2}]], \\ \mathbb{A}^{\mathrm{Re},n}\boldsymbol{\psi}^{n+1} = \mathbf{b}^{\mathrm{Re},n}, \end{cases} \tag{3.27}$$

where $\boldsymbol{\psi}^{n+1} = (\psi^{n+1}(\mathbf{x}_{j,k}))_{(j,k)\in\mathcal{P}_{J,K}}$ is a discrete unknown array in $\mathcal{M}_M(\mathbb{C})$ and $\mathbf{x}_{j,k} = (x_j, y_k)$. Here, $\mathcal{M}_M(\mathbb{C})$ designates the set of complex-valued 2d (respectively 1d and 3d) arrays, with $M = JK$ (respectively $M = J$ and $M = JKL$) in 2d (respectively 1d and 3d). For conciseness, let us remark that we do not make any distinction between an array $\boldsymbol{\phi}$ in $\mathcal{M}_M(\mathbb{C})$ and the corresponding reshaped vector in $\mathbb{C}^M$. The operator $\mathbb{A}^{\mathrm{Re},n}$ is a map from $\mathcal{M}_M(\mathbb{C})$ to itself such that

$$\mathbb{A}^{\mathrm{Re},n} := \left(i\frac{[[I]]}{\delta t} + \frac{1}{4}[[\Delta]] - \frac{1}{2}[[V^{n+1}]] - \frac{1}{2}[[\boldsymbol{\phi}^{n+1/2}]] + \frac{1}{2}\Omega[[L_z]]\right). \tag{3.28}$$

Moreover, the vector $\mathbf{b}^{\mathrm{Re},n}$ is given by

$$\mathbf{b}^{\mathrm{Re},n} := \left(i\frac{[[I]]}{\delta t} - \frac{1}{4}[[\Delta]] + \frac{1}{2}[[V^n]] + \frac{1}{2}[[\boldsymbol{\phi}^{n+1/2}]] - \frac{1}{2}\Omega[[L_z]]\right)\boldsymbol{\psi}^n. \tag{3.29}$$

The application of the operator $\mathbb{A}^{\mathrm{Re},n}$ and the evaluation of the vector $\mathbf{b}^{\mathrm{Re},n}$ are realized by applying the discretized operators. For the identity, the potential and the nonlinear operators, the application is direct since it is done pointwise in the physical space by setting

$$\forall (j,k)\in\mathcal{P}_{J,K}, \quad [[I]]_{j,k} := \delta_{j,k}, \qquad [[V]]_{j,k} := V(\mathbf{x}_{j,k}), \qquad [[|\boldsymbol{\psi}^n|^2]]_{j,k} = |\boldsymbol{\psi}^n(\mathbf{x}_{j,k})|^2. \tag{3.30}$$

The symbol $\delta_{j,k}$ denotes the Dirac delta symbol which is equal to 1 if and only if $j = k$ and 0 otherwise. By using (3.10) and (3.11), the partial differential operators in the x- and y-directions are discretized as

$$\forall (j,k)\in\mathcal{P}_{J,K}, \quad ([[\partial_x]]\boldsymbol{\psi})_{j,k} = \frac{1}{J}\sum_{p=-J/2}^{J/2-1} i\mu_p\widehat{\psi_p}(y_k,t)e^{i\mu_p(x_j+a_x)}, \quad ([[\partial_y]]\boldsymbol{\psi})_{j,k} = \frac{1}{K}\sum_{q=-K/2}^{K/2-1} i\lambda_q\widehat{\psi_q}(x_k,t)e^{i\lambda_q(y_k+a_y)},$$

leading to the following pseudospectral approximation of the operator L_z

$$\forall (j,k)\in\mathcal{P}_{J,K}, ([[L_z]]\boldsymbol{\psi})_{j,k} = -i\left(x_j([[\partial_y]]\boldsymbol{\psi})_{j,k} - y_k([[\partial_x]]\boldsymbol{\psi})_{j,k}\right). \tag{3.31}$$

The discrete second-order differential operators in the x- or y-directions are obtained through

$$\forall (j,k)\in\mathcal{P}_{J,K}, \quad ([[\partial_x^2]]\boldsymbol{\psi})_{j,k} = \frac{1}{J}\sum_{p=-J/2}^{J/2-1} -\mu_p^2\widehat{\psi_p}(y_k,t)e^{i\mu_p(x_j+a_x)}, \quad ([[\partial_y^2]]\boldsymbol{\psi})_{j,k} = \frac{1}{K}\sum_{q=-K/2}^{K/2-1} -\lambda_q^2\widehat{\psi_q}(x_k,t)e^{i\lambda_q(y_k+a_y)},$$

yielding the discrete Laplace operator Δ defined by

$$([[\Delta]]\boldsymbol{\psi})_{j,k} = \left([[\partial_x^2]]\boldsymbol{\psi} + [[\partial_y^2]]\boldsymbol{\psi}\right)_{j,k}. \tag{3.32}$$

We remark that the operator $[[\Delta]]$ is diagonal in the Fourier space but not $[[L_z]]$. The evaluation of a partial differential operator is made through FFT/iFFTs while the diagonal matrices in the physical space are directly applied. The linear system in (3.27) is solved at each time step by the BiCGStab or GMRES Krylov subspace solvers. In the spirit of [8, 10], an analytical preconditioner is used to accelerate the convergence of the iterative Krylov subspace solvers.

The resulting scheme is called Relaxation pseudo SPectral scheme (ReSP). The scheme is second-order in time and spectrally accurate in space. Its computational cost is $O(M \log M)$. The extension to the three-dimensional case and other nonlinearities is direct in terms of coding. Other properties are related to the fact that it is time reversible, mass and energy (for a cubic nonlinearity) conserving when the property holds at the continuous level. It is not time transverse invariant and the dispersion relation is not satisfied. The scheme is unconditionally stable (see [7] for more details).

3.2.2. Extension of the ReSP scheme to the multi-components case

The relaxation scheme can be extended to the multi-components situation in a similar way as in [8]. We consider the same notations as in section 3.1.3. We have the following time discretization of system (3.17) based on the relaxation scheme (for a general nonlinearity)

$$\left\{\begin{array}{l} \dfrac{\Phi^{n+1/2}+\Phi^{n-1/2}}{2} = \beta F(\Psi^n), \\ \dfrac{\Psi^{n+1}-\Psi^n}{\delta t} = -i\left(-\dfrac{1}{2}\Delta + \displaystyle\sum_{j=1}^{d} G^j \partial_{x_j} + \Phi^{n+1/2}\right)\dfrac{\Psi^{n+1}+\Psi^n}{2} + \dfrac{V^{n+1}\Psi^{n+1}+V^n\Psi^n}{2}, \end{array}\right.$$

for any $n \geq 0$. Concerning the spatial discretization, we use again a pseudo spectral method based on the FFTs/iFFTs. For the 2d case, the ReSP scheme reads

$$\left\{\begin{array}{l} [[\mathbf{\Phi}^{n+1/2}]] = 2\beta[[F(\mathbf{\Psi}^n)]] - [[\mathbf{\Phi}^{n-1/2}]], \\ \mathbb{A}^{\mathrm{Re},n}\mathbf{\Psi}^{n+1} = \mathbf{B}^{\mathrm{Re},n}, \end{array}\right. \tag{3.33}$$

where $\mathbf{\Psi}^n = ((\psi_1^n(\mathbf{x}_{j,k}))_{(j,k)\in\mathcal{P}_{J,K}}, ..., (\psi_{N_c}^n(\mathbf{x}_{j,k}))_{(j,k)\in\mathcal{P}_{J,K}})$ is the discrete unknown array in $\mathbb{C}^{MN_c}$, with $M := JK$. The relaxation operator $[[\mathbf{\Phi}^{n+1/2}]] \in \mathcal{M}_{MN_c}(\mathbb{C})$ is updated by computing the nonlinear operator

$$[[F(\mathbf{\Psi}^n)]] := \begin{pmatrix} [[F_{11}(\mathbf{\Psi}^n)]] & [[F_{12}(\mathbf{\Psi}^n)]] & \cdots & [[F_{1N_c}(\mathbf{\Psi}^n)]] \\ [[F_{21}(\mathbf{\Psi}^n)]] & [[F_{22}(\mathbf{\Psi}^n)]] & \cdots & [[F_{2N_c}(\mathbf{\Psi}^n)]] \\ \vdots & \vdots & \ddots & \vdots \\ [[F_{N_c1}(\mathbf{\Psi}^n)]] & [[F_{2N_c}(\mathbf{\Psi}^n)]] & \cdots & [[F_{N_cN_c}(\mathbf{\Psi}^n)]] \end{pmatrix},$$

setting $[[F_{\ell m}(\mathbf{\Psi}^n)]] = (F_{\ell m}(\mathbf{\Psi}^n(\mathbf{x}_{j,k})))_{(j,k)\in\mathcal{O}_{J,K}}$, $1 \leq \ell, m \leq N_c$. To be consistent with the one-component case, we consider: $[[\mathbf{\Phi}^{-1/2}]] = \beta[[F(\mathbf{\Psi}_0(\mathbf{x}))]]$. The operator $\mathbb{A}^{\mathrm{Re},n} \in \mathcal{M}_{MN_c}(\mathbb{C})$ is defined by

$$\mathbb{A}^{\mathrm{Re},n} := \left(i\frac{[[I_{N_c}]]}{\delta t} + \frac{1}{4}[[\Delta]] - \frac{1}{2}[[G^1]][[\partial_x]] - \frac{1}{2}[[G^2]][[\partial_y]] - \frac{1}{2}[[V^{n+1}]] - \frac{1}{2}[[\mathbf{\Phi}^{n+1/2}]]\right). \tag{3.34}$$

The vector $\mathbf{b}^{\mathrm{Re},n}$ is

$$\mathbf{b}^{\mathrm{Re},n} := \left(i\frac{[[I_{N_c}]]}{\delta t} - \frac{1}{4}[[\Delta]] + \frac{1}{2}[[G^1]][[\partial_x]] + \frac{1}{2}[[G^2]][[\partial_y]] + \frac{1}{2}[[V^n]] + \frac{1}{2}[[\mathbf{\Phi}^{n+1/2}]]\right)\mathbf{\Psi}^n. \tag{3.35}$$

The identity and potential operators are explicitly given by the matrices

$$[[I_{N_c}]] := \begin{pmatrix} [[I]] & 0 & \cdots & 0 \\ 0 & [[I]] & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & [[I]] \end{pmatrix} \in \mathcal{M}_{MN_c}(\mathbb{R}) \quad \text{and} \quad [[V]] := \begin{pmatrix} [[V_{11}]] & [[V_{12}]] & \cdots & [[V_{1N_c}]] \\ [[V_{21}]] & [[V_{22}]] & \cdots & [[V_{2N_c}]] \\ \vdots & \vdots & \ddots & \vdots \\ [[V_{N_c1}]] & [[V_{2N_c}]] & \cdots & [[V_{N_cN_c}]] \end{pmatrix} \in \mathcal{M}_{MN_c}(\mathbb{R}),$$

where the diagonal matrices are defined by: $[[I]]_{\ell m} = (\delta_{j,k})_{(j,k)\in\mathcal{O}_{J,K}} \in \mathcal{M}_M(\mathbb{R})$ and $[[V_{\ell m}]] = (V_{\ell m}(\mathbf{x}_{j,k}))_{(j,k)\in\mathcal{O}_{J,K}} \in \mathcal{M}_M(\mathbb{R})$. In (3.34) and (3.35) , we also have: $[[\Delta]]\mathbf{\Psi} := ([[\Delta\psi_\ell]])_{\ell=1,\dots,N_c} \in \mathbb{C}^{MN_c}$, and

$$[[\partial_x]]\mathbf{\Psi} := ([[\partial_x\psi_\ell]])_{\ell=1,\dots,N_c} \in \mathbb{C}^{MN_c}, \quad [[\partial_y]]\mathbf{\Psi} := \Big([[\partial_y\psi_\ell]]\Big)_{\ell=1,\dots,N_c} \in \mathbb{C}^{MN_c}. \tag{3.36}$$

For $k = 1, 2$, we define

$$[[G^k]] := \begin{pmatrix} [[G^k_{11}]] & [[G^k_{12}]] & \cdots & [[G^k_{1N_c}]] \\ [[G^k_{21}]] & [[G^k_{22}]] & \cdots & [[G^k_{2N_c}]] \\ \vdots & \vdots & \ddots & \vdots \\ [[G^k_{N_c1}]] & [[G^k_{2N_c}]] & \cdots & [[G^k_{N_cN_c}]] \end{pmatrix} \in \mathcal{M}_{MN_c}(\mathbb{C}),$$

setting $[[G^k_{\ell m}]] = (G^k_{\ell m}(\mathbf{x}_{j,k}))_{(j,k)\in\mathcal{O}_{J,K}} \in \mathcal{M}_M(\mathbb{C})$.

For solving the second equation of (3.33), we again use the preconditioned BiCGStab or GMRES. Concerning the TF(-like) preconditioner, since we have some coupling effects between the gazes through $[[V^{n+1}]]$ and $[[\mathbf{\Phi}^{n+1/2}]]$, the preconditioner is not diagonal. In GPELab, we propose to only keep the diagonal part for preconditioning, that is, to include the potential and nonlinear self-interactions in each gas. Concretely, we build the following *diagonal* TF preconditioner $\mathbb{P}^{\mathrm{Re},n}_{\mathrm{TF,diag}}$ given by

$$\mathbb{P}^{\mathrm{Re},n}_{\mathrm{TF,diag}} := \left(i\frac{[[I_{N_c}]]}{\delta t} - \frac{1}{2}[[V^{n+1}]]_{\mathrm{diag}} - \frac{1}{2}[[\mathbf{\Phi}^{n+1/2}]]_{\mathrm{diag}}\right)^{-1},$$

where $[[V^{n+1}]]_{\mathrm{diag}} := ([[V_{\ell\ell}]])_{\ell=1,\dots,N_c}$ and $[[\mathbf{\Phi}^{n+1/2}]]_{\mathrm{diag}} := ([[\mathbf{\Phi}^{n+1/2}]]_{\ell\ell})_{\ell=1,\dots,N_c}$. We can also consider the Laplace preconditioner $\mathbb{P}^{\mathrm{Re}}_{\Delta}$ which is built by inverting the laplacian in the Fourier space

$$\mathbb{P}^{\mathrm{Re}}_{\Delta} := \left(i\frac{[[I_{N_c}]]}{\delta t} + \frac{1}{4}[[\Delta]]\right)^{-1}.$$

3.3. Integration of a stochastic potential

GPELab offers the possibility of integrating a stochastic time-dependent potential in the GPE

$$V(t, \mathbf{x}) = V(\mathbf{x})\dot{w}_t,$$

where $(\dot{w}_t)_{t\geq 0}$ is a noise, i.e. the formal time derivative of a stochastic process $(w_t)_{t\geq 0} \in C^\gamma(\mathbb{R}^+)$, with $\gamma \in]0, 1[$. Let us explain how this class of potentials is discretized in the TSSP and ReSP schemes.

3.3.1. The case of the TSSP scheme

In the TSSP scheme, we essentially split the equation in two equations that we solve separately. In particular, the second step (see Eq. (3.5)) consists now in solving

$$\begin{cases} i\partial_t\psi_2(t, \mathbf{x}) = V(\dot{w}(t), \mathbf{x})\psi_2(t, \mathbf{x}) + \beta|\psi_2|^2\psi_2(t, \mathbf{x}), \ t_n < t \leq t_{n+1}, \forall \mathbf{x} \in \mathbb{R}^2, \\ \psi_2(t_n, \cdot) = \psi_1(t_{n+1}, \cdot). \end{cases} \tag{3.37}$$

We have seen in section 3.1.1 that we can exactly integrate the nonlinearity and the time-dependent potential for the deterministic case. Here, we obtain

$$t_n < t \leq t_{n+1}, \quad \psi_2(t, \mathbf{x}) = \psi_1(t_{n+1}, \mathbf{x})e^{-i\beta|\psi_1(t_{n+1},\mathbf{x})|^2(t-t_n) - i\int_{t_n}^t V(\mathbf{x})\dot{w}_s ds}.$$

Since the time integration of the stochastic potential gives

$$\int_{t_n}^t V(\mathbf{x})\dot{w}_s ds = V(\mathbf{x})(w_t - w_{t_n}),$$

one gets

$$\forall t \in [t_n, t_{n+1}], \quad \psi_2(t, \mathbf{x}) = \psi_1(t_{n+1}, \mathbf{x})e^{-i\beta|\psi_1(t_{n+1},\mathbf{x})|^2(t-t_n) - iV(\mathbf{x})(w_t - w_{t_n})}.$$

3.3.2. The case of the ReSP scheme

For the ReSP, we have to discretize the noise. To this end, we use the scheme

$$\int_{t_n}^{t_{n+1}} V(\mathbf{x})\psi(s,\mathbf{x})\dot{w}_s ds = V(\mathbf{x})\int_{t_n}^{t_{n+1}} \psi(s,\mathbf{x})\dot{w}_s ds \approx V(\mathbf{x})\frac{\psi(t_{n+1},\mathbf{x})+\psi(t_n,\mathbf{x})}{2}(w_{t_{n+1}}-w_{t_n}).$$

This leads to the ReSP scheme for the stochastic GPE

$$\begin{cases} \dfrac{\phi^{n+1/2}+\phi^{n-1/2}}{2} = \beta|\psi^n|^2, \\ i\dfrac{\psi^{n+1}-\psi^n}{\delta t} = (-\dfrac{1}{2}\Delta - \Omega L_z + V^n)\left(\dfrac{\psi^{n+1}+\psi^n}{2}\right) + \phi^{n+1/2}\left(\dfrac{\psi^{n+1}+\psi^n}{2}\right), \end{cases} \tag{3.38}$$

where $\phi^{n+1/2} = \phi(t_{n+1/2},\mathbf{x})$, $\psi^n = \psi(t_n,\mathbf{x})$ and $V^n = V(\mathbf{x})\left(\frac{w_{t_{n+1}}-w_{t_n}}{\delta t}\right)$.

4. GPELab functions for the dynamics

In [8], we presented the GPELab functions for computing the ground states of the GPE. In a similar way, functions are built for the dynamics. Compared to the stationary states computation, the numerical simulation of the dynamics remains close in terms of coding. This is first realized by using the `Method_Vard2d` and `Geometry2D_Var2d` functions. The `Geometry2D_Var2d` function is used in the same way as for stationary states [8]. Calling the `Method_Var2d` function is also similar but we nevertheless precise here the specific arguments for the dynamics. To avoid too much details, we refer to [8] for the various notations and an introduction to GPELab (section 7).

4.1. The `Method_Var2d` function

```
Method = Method_Var2d(Computation,Ncomponents, Type, Deltat, Stop_time, Stop_crit,
Max_iter, Precond_type, Output, Splitting, BESP, Solver_FD, Iterative_tol,
Iterative_maxit);
```

Table 1. The `Method_Var2d` function.

The `Method_Var2d` function creates the `Method` *structure* that contains all the parameters relative to the method (see [8] section 7.2.1 for the stationary case). Here, we specify the optional arguments for the dynamics

- `Computation` ($\mathbb{S}$,`'Dynamic'`) is a variable that must be `'Dynamic'` to compute the dynamic of the GPE (it can also be set to `'Ground'` for computing stationary states).
- `Ncomponents` ($\mathbb{N}$,1) is a variable corresponding to the number of components of the condensate.
- `Type` ($\mathbb{S}$, `'Splitting'`) is a variable which precises the scheme that is used in the simulation. In the case of a dynamical computation, it must be either `'Splitting'` for a TSSP scheme (see section 3.1) or `'Relaxation'` for the ReSP scheme (see section 3.2).
- `Deltat` ($\mathbb{R}^+$,`1e-3`) is the uniform time step.
- `Stop_time` ($\mathbb{R}^+$,1) is a variable that defines the final time of computation for the dynamics. The total number of time iterations is therefore
$$\#\text{Iter} = \text{Int}[[\frac{\texttt{Stop\_time}}{\texttt{Deltat}}]],$$
where Int[[·]] denotes the integer part function.
- `Stop_crit` ($\mathbb{R}^+$,`1e-8`) is a variable fixing the stopping criterion for the ground state computation [8].
- `Max_iter` ($\mathbb{N}$, `1e6`) is a variable corresponding to the maximum number of iterations for a stationary state computation.

- `Preconditioner` ($\mathbb{S}$,`'FLaplace'`) is a variable that must be either `'None'` for a calculation without preconditioner, `'Laplace'` for the Laplace preconditioner, `'ThomasFermi'` for the Thomas-Fermi preconditioner, `'FThomasFermi'` for a multi-components Thomas-Fermi preconditioner and `'FLaplace'` for a multi-components Laplace preconditioner.
- `Output` ($\mathbb{N}$,1) is a variable equals to 1 if one computes some outputs during the simulation or 0 otherwise.
- `Splitting` ($\mathbb{S}$,`'Strang'`) is a variable with the value `'Lie'` for the Lie TSSP scheme, `'Strang'` for the Strang TSSP scheme (see Section 3.1) or `'Fourth'` for a fourth-order TSSP scheme [11].
- `BESP` ($\mathbb{N}$,0) is a variable that must be either 1 if one uses a fixed-point Jacobi method or 0 for the Krylov method when using the BESP spectral scheme [8] (stationary case).
- `Solver_FD` ($\mathbb{N}$,0) is a variable equal to 1 when using a direct Gauss solver or 0 for the Krylov subspace method when a finite difference scheme (in space) is used [8] (stationary states).
- `Iterative_tol` ($\mathbb{R}^+$, `1e-9`) is a variable that fixes the residual stopping criterion in the Krylov subspace solver.
- `Iterative_maxit` ($\mathbb{N}$,`1e3`) is a variable that corresponds to the maximum number of iterations for the Krylov subspace solver.

Let us consider for example that we want to compute the time-dependent solution of a single-component BEC by using a TSSP scheme. We fix: $\delta t = 10^{-3}$ and a final time of computation $T = 1$. Moreover, we require that some outputs are calculated during the simulation. The resulting GPELab code is given in Table 2. (We remark that adding the preconditioner as `'Laplace'` does not change anything in the simulation since we use a (explicit) TSSP scheme.)

```
Computation = 'Dynamic;
Ncomponents = 1;
Type = 'Splitting';
Deltat = 1e-3;
Stop_time = 1;
Stop_crit = [];
Max_iter = [];
Precond_type = 'Laplace';
Output = 1;
Method = Method_Var2d(Computation,Ncomponents, Type, Deltat, Stop_time, Stop_crit,
Max_iter, Precond_type, Output);
```

Table 2. An example of initialization and use of the `Method_Var2d` function.

To set the physical problem, we need to define the operators involved in the GPE similarly to [8]. For the sake of completeness, we only list in sections 4.2 and 4.3 the physical operators which are specific to the dynamics.

4.2. The `TimePotential_Var2d` function

```
Physics2D = TimePotential_Var2d(Method, Physics2D, TimePotential, G );
```

Table 3. The `TimePotential_Var2d` function.

GPELab, through the `TimePotential_Var2d` function, allows to define a time-dependent potential operator (i.e. $\mathbf{V}(t,\mathbf{x})$) in the problem by modifying the `Physics2D` structure. It must be provided with the `Method` and `Physics2D` structures and takes the following optional arguments

- `TimePotential`: If a function `TimePotential` in $\mathbb{F}(\mathbb{R}^+, \mathcal{M}_{N_y,N_x}(\mathbb{R})^2; \mathcal{M}_{N_y,N_x}(\mathbb{C}))$ is provided, the physical time-dependent potential is defined as follows, for each $j,k \in \{1,...,N_c\}$,

$$\mathbf{V}_{j,k}(t,x,y) = \begin{cases} \texttt{TimePotential}(t,x,y) \text{ if } j = k \\ 0 \text{ if } j \neq k \end{cases}.$$

  If `TimePotential` is a cell array of functions in

$$C_{N_c,N_c}\{\mathbb{F}(\mathbb{R}^+, \mathcal{M}_{N_y,N_x}(\mathbb{R})^2; \mathcal{M}_{N_y,N_x}(\mathbb{C}))\},$$

  then the potential is

$$\mathbf{V}_{j,k}(t,x,y) = \texttt{TimePotential}\{j,k\}(t,x,y)$$

  for $j,k \in \{1,...,N_c\}$. The default argument is `quadratic_potential2d` which corresponds to

$$\mathbf{V}_{j,k}(t,x,y) = \begin{cases} \frac{1}{2}(x^2+y^2) \text{ if } j = k \\ 0 \text{ if } j \neq k \end{cases}$$

- `G` ($\mathcal{M}_{N_c,N_c}(\mathbb{C})$, `ones(N_c)`) is a complex-valued variable that multiplies the potential element-by-element, leading to the following time-dependent potential

$$\mathbf{V}_{j,k}(t,x,y) = \texttt{G}(j,k)\texttt{TimePotential}\{j,k\}(t,x,y)$$

  for $j,k \in \{1,...,N_c\}$.

To define a quadratic potential with a time-dependent intensity (see Table 4): $V(t,\mathbf{x}) = (\frac{1}{2} + \cos(t))|\mathbf{x}|^2$, its expression is first given and the `Physics2D` structure is modified through the `TimePotential_Var2d` function call.

```
Example_Timepotential = @(t,x,y) (1/2+cos(t)).*(x.^2 + y.^2)
Physics2D = TimePotential_Var2d(Method, Physics2D, Example_Timepotential );
```

Table 4. An example of how to use the `TimePotential_Var2d` function.

4.3. The `StochasticPotential_Var2d` function

```
Physics2D = StochasticPotential_Var2d(Method, Physics2D, StochasticPotential, G ,
StochasticProcess);
```

Table 5. The `StochasticPotential_Var2d` function.

It is possible to include a stochastic potential operator in the physical problem, i.e. a potential defined by a noise $\dot{w}_t$. The `StochasticPotential_Var2d` function defines the stochastic potential operator (i.e. $\mathbf{V}(\dot{w}_t, \mathbf{x})$) in the problem by modifying the `Physics2D` structure. It must be provided with the `Method` and `Physics2D` structures and considers the optional arguments

- `StochasticPotential`: If a function `StochasticPotential` in $\mathbb{F}(\mathbb{R}^+, \mathcal{M}_{N_y,N_x}(\mathbb{R})^2; \mathcal{M}_{N_y,N_x}(\mathbb{C}))$ is given, the physical stochastic potential is defined as follows, for each $j,k \in \{1,...,N_c\}$,

$$\mathbf{V}_{j,k}(\dot{w}_t,x,y) = \begin{cases} \texttt{StochasticPotential}(\dot{w}(t),x,y) \text{ if } j = k \\ 0 \text{ if } j \neq k \end{cases}$$

  If `StochasticPotential` is a cell array of functions in

$$C_{N_c,N_c}\{\mathbb{F}(\mathbb{R}^+, \mathcal{M}_{N_y,N_x}(\mathbb{R})^2; \mathcal{M}_{N_y,N_x}(\mathbb{C}))\},$$

then the potential is

$$\mathbf{V}_{j,k}(\dot{w}_t, x, y) = \texttt{StochasticPotential}\{j,k\}(\dot{w}_t, x, y)$$

for $j, k \in \{1, ..., N_c\}$. The default argument is `quadratic_potential2d`

$$\mathbf{V}_{j,k}(\dot{w}_t, x, y) = \begin{cases} \frac{1}{2}(x^2 + y^2) \text{ if } j = k \\ 0 \text{ if } j \neq k \end{cases}.$$

- `G` ($\mathcal{M}_{N_c,N_c}(\mathbb{C})$, `ones(N_c)`) is a complex-valued variable that multiplies each component of the potential

$$\mathbf{V}_{j,k}(\dot{w}_t, x, y) = \texttt{G}(j,k)\texttt{StochasticPotential}\{j,k\}(\dot{w}_t, x, y)$$

for $j, k \in \{1, ..., N_c\}$.

- `StochasticProcess` is a function corresponding to $(w_t)_{t\in\mathbb{R}^+}$ when defining the `StochasticPotential` function. The `StochasticPotential` function is computed by using a scalar value corresponding to $\dot{w}_t$.

For example, to define $V(\dot{w}_t, \mathbf{x}) = 1/2(x^2\dot{w}_t + y^2)$, where $(w_t)_{t\in\mathbb{R}^+}$ is a brownian motion, we first compute a brownian motion by using the `Brownian_Process2d` function which generates such a stochastic process. We provide the resulting GPELab code in Table 4 where we simulate a brownian motion and then we modify the `Physics2D` structure by using the `StochasticPotential_Var2d` function.

```
Brownian = Brownian_Process2d(Method);
Physics2D = StochasticPotential_Var2d(Method, Physics2D, ...
@(w,x,y) (1/2)*(x.^2*w + y.^2), [] , Brownian );
```

Table 6. How to use the `StochasticPotential_Var2d` function.

4.4. Further options before launching the simulation

The initial data can be defined by using the `InitialData_Var2d` function but can also be the result of the numerical computation of a stationary state by using GPELab. The outputs of a calculation are fixed through the `OutputsINI_Var2d` function and are computed like for the stationary case (by using the variable `Evo_outputs`, see section 7.4.1 in [8], the number of iterations corresponding to the number of time steps). We also need to build the `Print` and `Figure` structures (by using the `Print_Var2d` and `Figure_Var2d` functions, respectively, see sections 7.4.2 and 7.2.3 in [8]). Finally, the `GPELab2d` function is called to launch the simulation. We refer to [8] (section 7) for further details about these functions and variables.

5. Examples of computations

5.1. Collision of two bright solitons for a system of GPEs in 1d

This first example consists in computing the dynamics of two bright solitons for a system of GPEs with coupled cubic nonlinearities in 1d [46]. We define the following system of GPE

$$\begin{cases} i\partial_t\psi_1(t,x) = -\dfrac{1}{2}\Delta\psi_1(t,x) - \left[\alpha_1|\psi_1(t,x)|^2 + (\alpha_1 + 2\alpha_2)|\psi_2(t,x)|^2\right]\psi_1(t,x),\ t > 0,\ \forall x \in \mathbb{R}, \\ i\partial_t\psi_2(t,x) = -\dfrac{1}{2}\Delta\psi_2(t,x) - \left[\alpha_1|\psi_2(t,x)|^2 + (\alpha_1 + 2\alpha_2)|\psi_1(t,x)|^2\right]\psi_2(t,x),\ t > 0,\ \forall x \in \mathbb{R}, \\ \psi_1(0,x) = \psi_{1,0}(x) \ \text{ and } \ \psi_2(0,x) = \psi_{2,0}(x), \end{cases}$$

with $\alpha_1 = 0.25$ and $\alpha_2 = -0.1965$. We begin by building the `Method` and `Geometry1D` structures. We consider the ReSP scheme for two components, a time step $\delta t := 10^{-2}$ and a final time of computation $T = 25$. The grid for the spectral method uses $2^{11}+1$ points on the interval $]-40, 40[$ (see Table 7). We know that the default dispersion operator is the Laplace operator and we only need to define the coupled nonlinearities. Therefore, we build the `Physics1D`

structure with the correct coefficients and then add the default dispersion and the nonlinear operator to the physics of the problem (see Table 8). To simulate a bright soliton, we start with the initial data

$$\psi_{1,0}(x) = \sqrt{\frac{2}{\alpha_1}} b_\ell \mathrm{sech}(b_\ell(x - x_\ell)) \exp\left(i\frac{c_\ell}{2}x + n_\ell\right) \text{ and } \psi_{2,0}(x) = \sqrt{\frac{2}{\alpha_1}} b_r \mathrm{sech}(b_r(x - x_r)) \exp\left(i\frac{c_r}{2}x + n_r\right),$$

for $b_\ell = \sqrt{n_\ell + \frac{c_\ell^2}{4}}$, $b_r = \sqrt{n_r + \frac{c_r^2}{4}}$, $c_\ell = 0.15$, $c_r = -0.15$, $n_\ell = 0.03$, $n_r = 0.1$, $x_\ell = -15$ and $x_r = 0$. We use the `Geometry1D` structure to obtain the mesh grid through the variable `X` and compute the initial data (see Table 9). We print out the informations related to our computation in the command window every 15 iterations and draw the square of the amplitude of the solution. In addition, we also compute the position of the soliton through the formula

$$< x >= \int_{\mathbb{R}} x|\psi(t,x)|^2 dx, \tag{5.39}$$

as an output with the name `'Position of the soliton'` every 10 iterations. Furthermore, we save the solution during the simulation. The resulting code is given in Table 10. At the end of the simulation, we obtain the informations about the soliton by using the `Outputs` structure (like the soliton position). We can use `Draw_Timesolution1d` to draw the evolution of the modulus of each solution and print out the positions by using the `plot` Matlab function (see Table 11). We report the trajectories of the solitons on figures 1(a)-1(b) (where a splitting of the second soliton after the collision occurs) and the position of each soliton on figures 1(c)-1(d).

```
Computation = 'Dynamic';
Ncomponents = 2;
Type = 'Relaxation';
Deltat = 1e-2;
Stop_time = 25;
Method = Method_Var1d(Computation,Ncomponents, Type, Deltat, Stop_time);
xmin = -40;
xmax = 40;
Nx = 2^11+1;
Geometry1D = Geometry1D_Var1d(xmin,xmax, Nx);
```

Table 7. Building the `Method` and `Geometry1D` structures to compute the initial data.

```
Delta = 1;
Beta = 1;
alpha_1 = 0.25;
alpha_2 = -0.1965;
Physics1D = Physics1D_Var1d(Method, Delta, Beta);
Physics1D = Dispersion_Var1d(Method, Physics1D);
Coupled_NL{1,1} = @(Phi,X) alpha_1*abs(Phi{1}).^2 + (alpha_1+2*alpha_2)*abs(Phi{2}).^2;
Coupled_NL{1,2} = @(Phi,X) 0;
Coupled_NL{2,1} = @(Phi,X) 0;
Coupled_NL{2,2} = @(Phi,X) alpha_1*abs(Phi{2}).^2 + (alpha_1+2*alpha_2)*abs(Phi{1}).^2;
Physics1D = Nonlinearity_Var1d(Method, Physics1D, Coupled_NL);
```

Table 8. Setting the `Physics1D` structure and adding the nonlinear operator.

5.2. Dynamics of a rotating Bose-Einstein condensate perturbed by a random gaussian potential in 2d

We consider now the case of the dynamics of a 2d single-component GPE with a cubic nonlinearity, a rotating operator and a random potential. The random potential is a gaussian potential with a random intensity which creates

```
c_l = 1.2;
n_l = 0.03;
b_l = sqrt(n_l+c_l^2/4);
X_l = -15;
X = Geometry1D.X;
Phi_0{1} = sqrt(2/abs(alpha_1))*b_l*sech(b_l*(X-X_l)).*exp(1i*c_l*X/2+ n_l);
c_r = -0.5;
n_r = 0.1;
b_r = sqrt(n_r+c_r^2/4);
X_r = 0;
X = Geometry1D.X;
Phi_0{2} = sqrt(2/abs(alpha_1))*b_r*sech(b_r*(X-X_r)).*exp(1i*c_r*X/2+ n_r);
```

Table 9. Building the initial data.

```
Solution_save = 1;
Outputs_iterations = 10;
Output_function{1} = @(Phi,X,FFTX) Geometry1D.dx*sum(X.*abs(Phi).^2);
Output_name{1} = 'Position of the soliton';
Outputs = OutputsINI_Var1d(Method,Outputs_iterations,Solution_save,Output_function,...
Output_name);
Printing = 1;
Evo = 15;
Draw = 1;
Print = Print_Var1d(Printing,Evo,Draw);
[Phi,Outputs]= GPELab1d(Phi_0,Method,Geometry1D,Physics1D,Outputs,[],Print);
```

Table 10. Setting the outputs and the `Print` structure then launching the simulation.

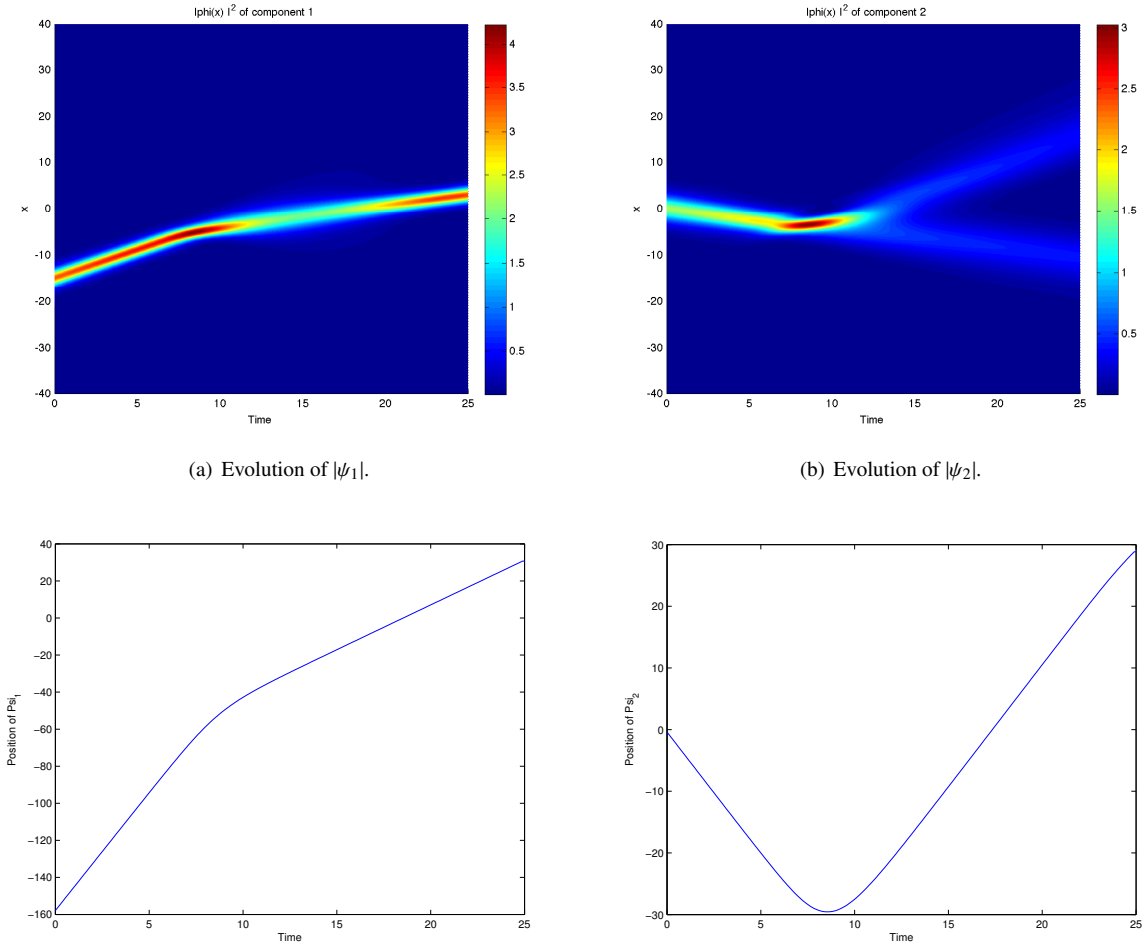
```
Draw_Timesolution1d(Outputs,Method,Geometry1D,Figure_Var1d);
figure(3)
Time = [0:0.1:25];
plot(Time, Outputs.User_defined_local{1,1})
xlabel('Time')
ylabel('Position of Psi_1')
figure(4)
plot(Time, Outputs.User_defined_local{2,1})
xlabel('Time')
ylabel('Position of Psi_2')
```

Table 11. Plotting the evolution of the soliton position.

sound waves in the condensate. Our goal is to observe the interaction between the sound waves and the vortices [45]. We first compute a ground state for the deterministic GPE with a quadratic potential and a cubic nonlinearity

$$i\partial_t\psi(x,y,t) = -\frac{1}{2}\Delta\psi(x,y,t) + \frac{1}{2}(|x|^2+|y|^2)\psi(x,y,t) + \beta|\psi(x,y,t)|^2\psi(x,y,t) + i\Omega(x\partial_y - y\partial_x)\psi(x,y,t),$$

with $\beta = 1000$ and $\Omega = 0.6$. We build the `Method` and `Geometry2D` structures and consider the BESP scheme [8, 10] for the stationary state computation with $\delta t = 0.5$ and the (weak) stopping criterion $\varepsilon = 10^{-8}$. The computational box is $\mathcal{O} =]-10, 10[^2$ for $J = K = 2^8$ grid points (see Table 12). Thanks to Table 13, the physics is built by using the `Potential_Var2d`, `Nonlinearity_Var2d`, `Gradientx_Var2d` and `Gradienty_Var2d` functions. The initial wave function is the Thomas-Fermi approximation which is fixed in GPELab by using the `InitialData_Var2d` function



(a) Evolution of $|\psi_1|$.

(b) Evolution of $|\psi_2|$.

(c) Position of ψ_1 during the simulation.

(d) Position of ψ_2 during the simulation.

Figure 1. Collision of two bright solitons for a system of GPEs in 1d.

(see Table 14). To launch the computation, the defaults `Outputs` (respectively `Print`) structure is first initialized by using the `OutputsINI_Var2d` (respectively `Print_Var2d`) function. Next, the `GPELab2d` function is called and the ground state is stored in the variable `Phi_1` (see Table 15). At the end of the computation, we obtain the ground state whose modulus is depicted in Figure 2.

We consider now a random gaussian potential in the GPE and simulate the nonlinear dynamics of the BEC. We first rebuild the `Method` structure for this dynamical problem. We use a TSSP scheme with $\delta t := 10^{-3}$ for a maximal time of computation $T = 1$ (see Table 16). The random gaussian potential is defined by

$$V(t, \boldsymbol{x}) = V_0 e^{-|\boldsymbol{x}-\boldsymbol{x}_0|^2/2\ell^2} \dot{w}_t,$$

where $(\dot{w}_t)_{t\in\mathbb{R}^+}$ is a white noise. We first compute a brownian motion $(w_t)_{t\in\mathbb{R}^+}$ and then add the stochastic potential to the `Physics2D` structure by using the `Brownian_Process2d` GPELab function (see Table 17). We next set the outputs, the printing informations (Table 18), and then launch the simulation. At the end of the simulation, we draw some snapshots of the modulus of the BEC and observe the propagation of sound waves in the rotating Bose-Einstein condensate (see Figures 3(a)-3(d)).

```
Computation = 'Ground';
Ncomponents = 1;
Type = 'BESP';
Deltat = 1e-1;
Stop_time = [];
Stop_crit = {'Energy',1e-8};
Method = Method_Var2d(Computation,Ncomponents, Type, Deltat, Stop_time , Stop_crit);
xmin = -10;
xmax = 10;
ymin = -10;
ymax = 10;
Nx = 2^8+1;
Ny = 2^8+1;
Geometry2D = Geometry2D_Var2d(xmin,xmax, ymin,ymax, Nx, Ny);
```

Table 12. Building the `Method` and `Geometry2D` structures for the computation of a stationary state.

```
Delta = 0.5;
Beta = 1000;
Omega = 0.52;
Physics2D = Physics2D_Var2d(Method,Delta,Beta,Omega);
Physics2D = Potential_Var2d(Method, Physics2D, @(x,y) (1/2)*(x.^2+y.^2));
Physics2D = Nonlinearity_Var2d(Method, Physics2D, @(phi,x,y) abs(phi).^2 );
Physics2D = Gradientx_Var2d(Method, Physics2D, @(x,y) -1i*Omega*y );
Physics2D = Gradienty_Var2d(Method, Physics2D, @(x,y) 1i*Omega*x );
```

Table 13. Setting the `Physics2D` structure to compute the stationary state.

```
InitialData_choice = 2 ;
Phi_0 = InitialData_Var2d(Method, Geometry2D, Physics2D,InitialData_choice);
```

Table 14. Initialization by the Thomas-Fermi approximation.

```
Outputs = OutputsINI_Var2d(Method);
Printing = 1;
Evo = 15;
Draw = 1;
Print = Print_Var2d(Printing,Evo,Draw);
[Phi_1,Outputs]= GPELab2d(Phi_0,Method,Geometry2D,Physics2D,Outputs,[],Print);
```

Table 15. Launching the computation of the ground state.

```
Computation = 'Dynamic';
Ncomponents = 1;
Type = 'Splitting';
Deltat = 1e-3;
Stop_time = 1;
Stop_crit = [];
Method = Method_Var2d(Computation,Ncomponents, Type, Deltat, Stop_time , Stop_crit);
```

Table 16. Building the `Method` structure for a dynamical problem.

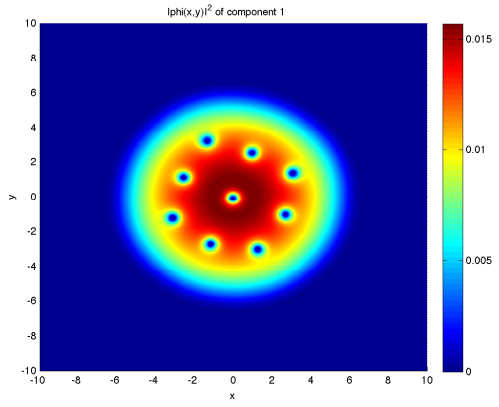


Figure 2. Ground state computed with GPELab by using the parameters from section 5.2.

```
X_0 = 0;
Y_0 = 0;
d = 4;
V_0 = 2;
Brownian = Brownian_Process2d(Method);
Physics2D = StochasticPotential_Var2d(Method, Physics2D, @(W,X,Y)
V_0*exp(-((X-X_0).^2+(Y-Y_0).^2)/2*d^2).*W, [], @(t,X,Y) Brownian(t));
```

Table 17. Adding the random potential.

```
Save_Solution = 1;
Outputs = OutputsINI_Var2d(Method,Save_Solution);
Printing = 1;
Evo = 10;
Draw = 1;
Print = Print_Var2d(Printing,Evo,Draw);
[Phi,Outputs]= GPELab2d(Phi_1,Method,Geometry2D,Physics2D,Outputs,[],Print);
```

Table 18. Printing instructions.

5.3. Dynamics of a superfluid with a random initial data in 3d

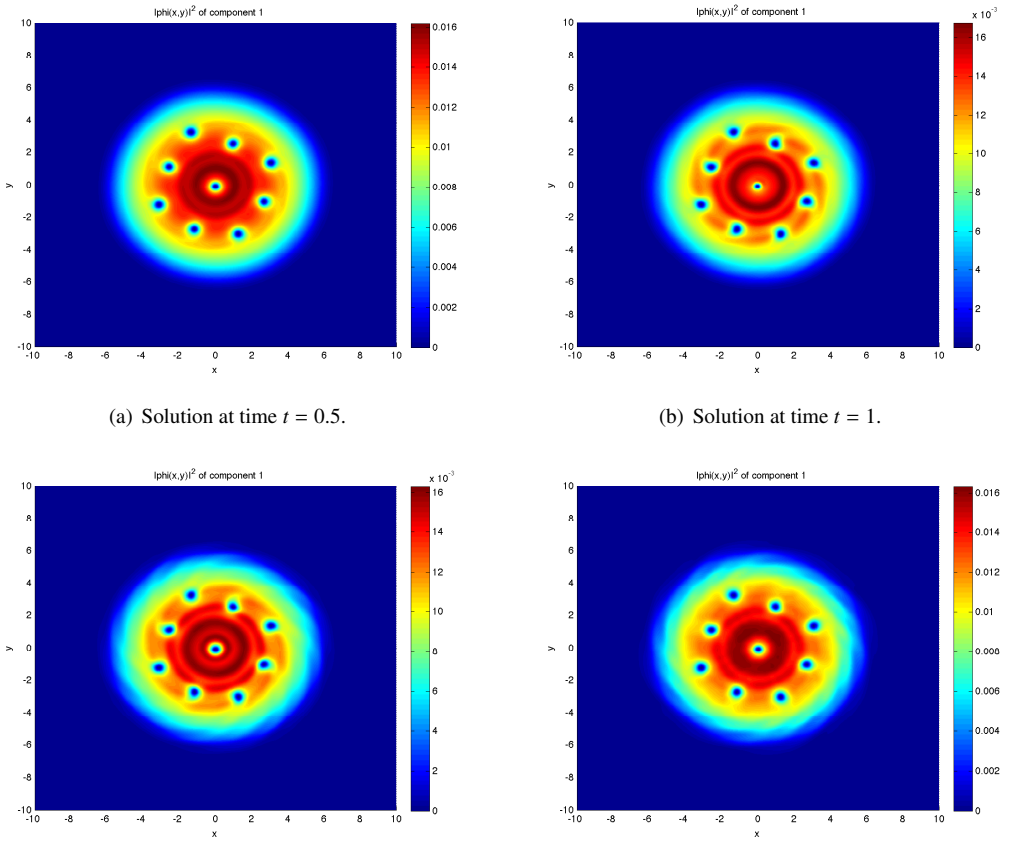
The third and last example is related to the nonlinear dynamics of a turbulent superfluid [28]. We want to simulate the dynamics of the following nonlinear Schrödinger equation

$$i\partial_t\psi(t,x,y,z) = -\frac{1}{2}\Delta\psi(t,x,y,z) + \beta|\psi(t,x,y,z)|^2\psi(t,x,y,z), t > 0, \tag{5.40}$$

with an initial data corresponding to a superfluid with a uniform density and a random phase. In this simulation, we consider a splitting scheme with a time step $\delta t = 10^{-3}$ and the computational domain $\mathcal{O} =]-2,2[^3$, with $J = K = L = 2^7$ grid points. We build the `Method` and `Geometry2D` structures (see Table 19). We consider now the 3d GPE (5.40) for $\beta = 0.001$ (see Table 20). Following a procedure similar to [28], we set the initial data as

$$\psi_0(x,y,z) = e^{i\phi(x,y,z)},$$

where ϕ is a random gaussian field with a covariance function given by $c(x,y,z) = ae^{-(x^2+y^2+z^2)/2d^2}$, with $a = 1$ and $d = 0.5$. We compute ϕ with the help of the `Stationary_Gaussian_Field3d` function (see Table 23). Then, we



(a) Solution at time $t = 0.5$.

(b) Solution at time $t = 1$.

(c) Solution at time $t = 1.5$.

(d) Solution at time $t = 2$.

Figure 3. Evolution of a Bose-Einstein condensate perturbed by a random gaussian potential.

define the `Outputs` structure so that we compute the mean momentum $< \boldsymbol{p}_j >_{j=1,2,3}$ of the BEC (see Table 24)

$$< \boldsymbol{p}_j >= \int_{\mathbb{R}^3} \psi(t,\boldsymbol{x})^* \partial_{\boldsymbol{x}_j} \psi(t,\boldsymbol{x}) d\boldsymbol{x}.$$

Finally, the `Print` structure is defined and the `GPELab3d` function is called. We print the informations every 10 iterations, draw the modulus (by using isovalues) and the phase (by using slices) (see Table 25). At the end of the simulation, we draw the 10^{-6}-isovalues of the modulus of the solution at time $T = 1$ without transparency (by setting `alpha = 1`) (see Table 26). This leads to Figures 4(a)-4(b) where, in particular, we observe some vortices filamentation.

6. Conclusion

This second paper presents the functionalities of GPELab to compute the deterministic and stochastic nonlinear dynamics of BECs modeled through the GPEs. GPELab includes some robust and efficient time-splitting and relaxation schemes in time with spectral accuracy in space (related to FFTs) for a large class of systems of GPEs. After presenting the specific GPELab functions for the nonlinear dynamics, a few 1d-2d-3d examples show how to build advanced GPELab scripts.

Acknowledgements. This work was partially supported by the French ANR grant MicroWave NT09_460489 (“Programme Blanc” call) and ANR-12-MONU-0007-02 BECASIM (Modèles Numériques call).

```
Computation = 'Dynamic';
Ncomponents = 1;
Type = 'Splitting';
Deltat = 1e-3;
Stop_time = 1;
Stop_crit = [];
Method = Method_Var3d(Computation,Ncomponents, Type, Deltat, Stop_time,Stop_crit);
xmin = -2;
xmax = 2;
ymin = -2;
ymax = 2;
zmin = -2;
zmax = 2;
Nx = 2^7+1;
Ny = 2^7+1;
Nz = 2^7+1;
Geometry3D = Geometry3D_Var3d(xmin,xmax, ymin,ymax, Nx, Ny);
```

Table 19. Building the `Method` and `Geometry3D` structures for the computation of the initial data.

```
Delta = 1;
Beta = 1e-3;
Physics3D = Physics3D_Var3d(Method,Delta,Beta);
Physics3D = Potential_Var3d(Method, Physics3D);
Physics3D = Nonlinearity_Var3d(Method, Physics3D);
```

Table 20. Building and defining the `Physics3D` structure for the initial data.

```
InitialData_choice = 2 ;
Phi_0 = InitialData_Var3d(Method, Geometry3D, Physics3D,InitialData_choice);
```

Table 21. Initialization by the Thomas-Fermi approximation.

```
Outputs_iterations = 10;
Outputs_save = 0:
Outputs = OutputsINI_Var3d(Method,Outputs_iterations,Outputs_save);
Printing = 1;
Evo = 15;
Draw = 1;
Print = Print_Var3d(Printing,Evo,Draw);
[Phi_1,Outputs]= GPELab3d(Phi_0,Method,Geometry3D,Physics3D,Outputs,[],Print);
```

Table 22. Launching the computation of a stationary state to provide an initial data.

```
A = 1;
d = 0.5;
Random_Phase = Stationary_Gaussian_Field3d(Geometry3D,...
@(X,Y,Z) A*exp(-(X.^2 + Y.^2 + Z.^2)/(2*d^2)));
Phi_1{1} = exp(-2i*pi*Random_Phase);
```

Table 23. Adding a random phase to the initial state.

```
Solution_save = 0;
Outputs_iterations = 10;
Output_function{1} = @(Phi,X,Y,Z,FFTX,FFTY,FFTZ) Geometry3D.dx*Geometry3D.dy*...
Geometry3D.dz*sum(sum(sum(ifftn(FFTX.*fftn(Phi)).*conj(Phi))));
Output_function{2} = @(Phi,X,Y,Z,FFTX,FFTY,FFTZ) Geometry3D.dx*Geometry3D.dy*...
Geometry3D.dz*sum(sum(sum(ifftn(FFTY.*fftn(Phi)).*conj(Phi))));
Output_function{3} = @(Phi,X,Y,Z,FFTX,FFTY,FFTZ) Geometry3D.dx*Geometry3D.dy*...
Geometry3D.dz*sum(sum(sum(ifftn(FFTZ.*fftn(Phi)).*conj(Phi))));
Output_name{1} = 'BEC Momentum X';
Output_name{2} = 'BEC Momentum Y';
Output_name{3} = 'BEC Momentum Z';
Outputs = OutputsINI_Var3d(Method,Outputs_iterations,Solution_save,...
Output_function,Output_name);
```

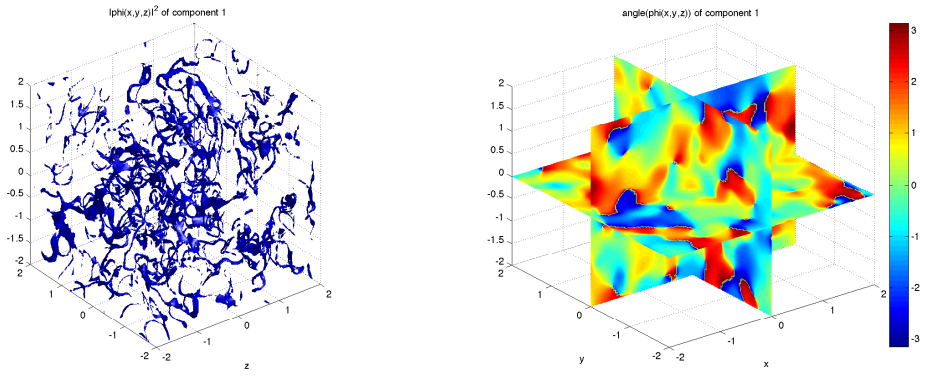
Table 24. Setting the `Outputs` structure for the dynamical problem.

```
Printing = 1;
Evo = 10;
Draw = 1;
Print = Print_Var3d(Printing,Evo,Draw);
[Phi,Outputs]= GPELab3d(Phi_1,Method,Geometry3D,Physics3D,Outputs,[],Print);
```

Table 25. Launching the simulation with the `GPELab3d` function.

```
View = 3;
Isovalue = 1e-6;
Aspect = 1;
Figure = Figure_Var3d(View,Isovalue,Aspect);
Draw_Solution3d(Phi,Method,Geometry3D,Figure);
```

Table 26. Drawing 10^{-6}-isovalues of the modulus with no transparency.



(a) 10^{-6}-isovalues of the modulus of the solution at time $t = 1$. (b) Slice of the phase of the solution at time $t = 1$.

Figure 4. Turbulence in a superfluid: modulus and phase of the solution at the end of the simulation.

References

[1] F. Kh. Abdullaev, B. B. Baizakov, and V. V. Konotop. Dynamics of a Bose-Einstein condensate in optical trap. In *Nonlinearity and Disorder: Theory and Applications*, volume 45 of *NATO Science Series*, pages 69–78. Springer Netherlands, 2001.

[2] F. Kh. Abdullaev, J. C. Bronski, and R. M. Galimzyanov. Dynamics of a trapped 2d Bose-Einstein condensate with periodically and randomly varying atomic scattering length. *Physica D: Nonlinear Phenomena*, 184(1-4):319 – 332, 2003.

[3] F. Kh. Abdullaev, J. C Bronski, and G. Papanicolaou. Soliton perturbations and the random Kepler problem. *Physica D: Nonlinear Phenomena*, 135(3-4):369 – 386, 2000.

[4] J. R. Abo-Shaeer, C. Raman, J. M. Vogels, and W. Ketterle. Observation of vortex lattices in Bose-Einstein condensates. *Science*, 292(5516):476–479, 2001.

[5] A. Aftalion and P. Mason. Phase diagrams and Thomas-Fermi estimates for spin-orbit-coupled Bose-Einstein condensates under rotation. *Phys. Rev. A*, 88:023610, Aug 2013.

[6] M. H. Anderson, J. R. Ensher, M. R. Matthews, C. E. Wieman, and E. A. Cornell. Observation of Bose-Einstein condensation in a dilute atomic vapor. *Science*, 269(5221):198–201, 1995.

[7] X. Antoine, W. Bao, and C. Besse. Computational methods for the dynamics of the nonlinear Schrödinger/Gross-Pitaevskii equations. *Computer Physics Communications*, 184(12):2621 – 2633, 2013.

[8] X. Antoine and R. Duboscq. GPELab, a Matlab toolbox to solve Gross-Pitaevskii equations I: Computation of stationary solutions. *Computer Physics Communications*, 185(11):2969–2991, 2014.

[9] X. Antoine and R. Duboscq. *Modeling and computation of Bose-Einstein condensates: stationary states, nucleation, dynamics, stochasticity*. Lecture Notes in Mathematics. Springer, to appear, 2014.

[10] X. Antoine and R. Duboscq. Robust and efficient preconditioned Krylov spectral solvers for computing the ground states of fast rotating and strongly interacting Bose-Einstein condensates. *Journal of Computational Physics*, 258C:509–523, 2014.

[11] W. Bao and Y. Cai. Mathematical theory and numerical methods for Bose-Einstein condensation. *Kinetic and Related Models*, 6(1):1–135, 2013.

[12] 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. *Journal of Computational Physics*, 219(2):836–854, 2006.

[13] W. Bao, Q. Du, and Y. Zhang. Dynamics of rotating Bose-Einstein condensates and its efficient and accurate numerical computation. *SIAM Journal on Applied Mathematics*, 66(3):758–786, 2006.

[14] W. Bao, D. Jaksch, and P. A. Markowich. Numerical solution of the Gross-Pitaevskii equation for Bose-Einstein condensation. *Journal of Computational Physics*, 187(1):318–342, 2003.

[15] W. Bao and H. Wang. An efficient and spectrally accurate numerical method for computing dynamics of rotating Bose-Einstein condensates. *Journal of Computational Physics*, 217(2):612–626, 2006.

[16] C. Besse. A relaxation scheme for the nonlinear Schrödinger equation. *SIAM Journal on Numerical Analysis*, 42(3):934–952, 2004.

[17] K. B. Davis, M-O. Mewes, M. R. van Andrews, N. J. Van Druten, D. S. Durfee, D. M. Kurn, and W. Ketterle. Bose-Einstein condensation in a gas of sodium atoms. *Physical Review Letters*, 75(22):3969–3973, 1995.

[18] D. G. Fried, T. C. Killian, L. Willmann, D. Landhuis, S. C. Moss, D. Kleppner, and T. J. Greytak. Bose-Einstein condensation of atomic hydrogen. *Physical Review Letters*, 81:3811–3814, Nov 1998.

[19] J. Garnier, F. Kh. Abdullaev, and B. B. Baizakov. Collapse of a Bose-Einstein condensate induced by fluctuations of the laser intensity. *Physical Review A*, 69:053607, May 2004.

[20] M. E. Gehm, K. M. O'Hara, T. A. Savard, and J. E. Thomas. Dynamics of noise-induced heating in atom traps. *Physical Review A*, 58:3914–3921, Nov 1998.

[21] S. Giovanazzi, A. Görlitz, and T. Pfau. Tuning the dipolar interaction in quantum gases. *Phys. Rev. Lett.*, 89:130401, Sep 2002.

[22] K. Góral, K. Rzążewski, and T. Pfau. Bose-Einstein condensation with magnetic dipole-dipole forces. *Physical Review A*, 61:051601, Mar 2000.

[23] A. Griesmaier, J. Werner, S. Hensler, J. Stuhler, and T. Pfau. Bose-Einstein condensation of chromium. *Physical Review Letters*, 94:160401, Apr 2005.

[24] E. P. Gross. Structure of a quantized vortex in boson systems. *Il Nuovo Cimento Series 10*, 20(3):454–477, 1961.

[25] B. Jackson, J. F. McCann, and C. S. Adams. Vortex formation in dilute inhomogeneous Bose-Einstein condensates. *Physical Review Letters*, 80:3903–3906, 1998.

[26] K. Kasamatsu, M. Tsubota, and M. Ueda. Giant hole and circular superflow in a fast rotating Bose-Einstein condensate. *Phys. Rev. A*, 66:053606, Nov 2002.

[27] Y. Kawaguchi and M. Ueda. Spinor Bose-Einstein condensates. *Physics Reports*, 2012.

[28] M. Kobayashi and M. Tsubota. Kolmogorov spectrum of superfluid turbulence: Numerical analysis of the Gross-Pitaevskii equation with a small-scale dissipation. *Physical review letters*, 94(6):065302, 2005.

[29] T. Kuga, Y. Torii, N. Shiokawa, T. Hirano, Y. Shimizu, and H. Sasada. Novel optical trap of atoms with a doughnut beam. *Phys. Rev. Lett.*, 78:4713–4716, Jun 1997.

[30] T. Lahaye, C. Menotti, L. Santos, M. Lewenstein, and T. Pfau. The physics of dipolar bosonic quantum gases. *Reports on Progress in Physics*, 72(12):126401, 2009.

[31] C. K. Law, H. Pu, and N. P. Bigelow. Quantum spins mixing in spinor Bose-Einstein condensates. *Phys. Rev. Lett.*, 81:5257–5261, Dec 1998.

[32] M. Lewin, P. T. Nam, and N. Rougerie. Derivation of Hartree's theory for generic mean-field Bose systems. *arXiv preprint arXiv:1303.0981*, 2013.

[33] E. H. Lieb and R. Seiringer. Derivation of the Gross-Pitaevskii equation for rotating Bose gases. *Communications in Mathematical Physics*, 264(2):505–537, 2006.

[34] K. W. Madison, F. Chevy, V. Bretin, and J. Dalibard. Stationary states of a rotating Bose-Einstein condensate: routes to vortex nucleation. *Physical Review Letters*, 86(20):4443–4446, 2001.

[35] K. W. Madison, F. Chevy, W. Wohlleben, and J. Dalibard. Vortex formation in a stirred Bose-Einstein condensate. *Physical Review Letters*, 84(5):806–809, 2000.

[36] K. W. Madison, F. Chevy, W. Wohlleben, and J. Dalibard. Vortices in a stirred Bose-Einstein condensate. *Journal of Modern Optics*, 47(14-15):2715–2723, 2000.

[37] H.-J. Miesner, D. M. Stamper-Kurn, J. Stenger, S. Inouye, A. P. Chikkatur, and W. Ketterle. Observation of metastable states in spinor Bose-Einstein condensates. *Phys. Rev. Lett.*, 82:2228–2231, Mar 1999.
[38] C. J. Myatt, E. A. Burt, R. W. Ghrist, E. A. Cornell, and C. E. Wieman. Production of two overlapping Bose-Einstein condensates by sympathetic cooling. *Physical Review Letters*, 78:586–589, Jan 1997.
[39] P. Pedri and L. Santos. Two-dimensional bright solitons in dipolar Bose-Einstein condensates. *Physical Review Letters*, 95:200404, Nov 2005.
[40] C. J. Pethick and H. Smith. *Bose-Einstein condensation in dilute gases*. Cambridge University Press, 2002.
[41] L. P. Pitaevskii. Vortex lines in an imperfect bose gas. *Soviet Physics JETP-USSR*, 13(2), 1961.
[42] C. Raman, J. R. Abo-Shaeer, J. M. Vogels, K. Xu, and W. Ketterle. Vortex nucleation in a stirred Bose-Einstein condensate. *Physical Review Letters*, 87(21):210402, 2001.
[43] L. E. Sadler, J. M. Higbie, S. R. Leslie, M. Vengalattore, and D. M. Stamper-Kurn. Spontaneous symmetry breaking in a quenched ferromagnetic spinor Bose-Einstein condensate. *Nature*, 443(7109):312–315, 2006.
[44] T. A. Savard, K. M. O'Hara, and J. E. Thomas. Laser-noise-induced heating in far-off resonance optical traps. *Physical Review A*, 56:R1095–R1098, Aug 1997.
[45] T. P. Simula, P. Engels, I. Coddington, V. Schweikhard, E. A. Cornell, and R. J. Ballagh. Observations on sound propagation in rapidly rotating Bose-Einstein condensates. *Physical Review Letters*, 94(8):080404, 2005.
[46] W. J. Sonnier and C. I. Christov. Repelling soliton collisions in coupled Schrödinger equations with negative cross modulation. *Discret. Contin. Dyn. S.*, pages 708–718, 2009.
[47] M. Thalhammer. Convergence analysis of high-order time-splitting pseudospectral methods for nonlinear Schrödinger equations. *SIAM Journal on Numerical Analysis*, 50(6):3231–3258, 2012.
[48] L. Wen, H. Xiong, and B. Wu. Hidden vortices in a Bose-Einstein condensate in a rotating double-well potential. *Physical Review A*, 82(5):053627, 2010.
[49] X.-Q. Xu and J. H. Han. Spin-orbit coupled Bose-Einstein condensate under rotation. *Phys. Rev. Lett.*, 107:200401, Nov 2011.
[50] R. Zeng and Y. Zhang. Efficiently computing vortex lattices in rapid rotating Bose-Einstein condensates. *Computer Physics Communications*, 180(6):854–860, 2009.