Computer Physics Communications 209 (2016) 144–162

Contents lists available at ScienceDirect

Computer Physics Communications

journal homepage: www.elsevier.com/locate/cpc

A finite-element toolbox for the stationary Gross–Pitaevskii equation with rotation☆

Guillaume Vergez [a,b], Ionut Danaila [a,*], Sylvain Auliac [b], Frédéric Hecht [b]

[a] Université de Rouen Normandie, Laboratoire de Mathématiques Raphaël Salem, CNRS UMR 6085, Avenue de l'Université, BP 12, F-76801 Saint-Étienne-du-Rouvray, France

[b] UPMC Univ Paris 06, CNRS UMR 7598, Laboratoire Jacques-Louis Lions, 4 Place Jussieu, F-75005 Paris, France

ARTICLE INFO

Article history:
Received 31 January 2016
Received in revised form
27 June 2016
Accepted 20 July 2016
Available online 13 September 2016

Keywords:
FreeFem++
Ipopt
Gross–Pitaevskii
Bose–Einstein
Finite element
Mesh adaptivity
Sobolev gradient

ABSTRACT

We present a new numerical system using classical finite elements with mesh adaptivity for computing stationary solutions of the Gross–Pitaevskii equation. The programs are written as a toolbox for FreeFem++ (www.freefem.org), a free finite-element software available for all existing operating systems. This offers the advantage to hide all technical issues related to the implementation of the finite element method, allowing to easily code various numerical algorithms. Two robust and optimized numerical methods were implemented to minimize the Gross–Pitaevskii energy: a steepest descent method based on Sobolev gradients and a minimization algorithm based on the state-of-the-art optimization library Ipopt. For both methods, mesh adaptivity strategies are used to reduce the computational time and increase the local spatial accuracy when vortices are present. Different run cases are made available for 2D and 3D configurations of Bose–Einstein condensates in rotation. An optional graphical user interface is also provided, allowing to easily run predefined cases or with user-defined parameter files. We also provide several post-processing tools (like the identification of quantized vortices) that could help in extracting physical features from the simulations. The toolbox is extremely versatile and can be easily adapted to deal with different physical models.

Program summary

Program title: GPFEM
Catalogue identifier: AFBD_v1_0
Program summary URL: http://cpc.cs.qub.ac.uk/summaries/AFBD_v1_0.html
Program obtainable from: CPC Program Library, Queen's University, Belfast, N. Ireland
Licensing provisions: Apache 2.0
No. of lines in distributed program, including test data, etc.: 49149
No. of bytes in distributed program, including test data, etc.: 407572
Distribution format: tar.gz
Programming language: FreeFem++ (free software, www.freefem.org).
Computer: PC, Mac, Super-computer.
Operating system: Windows, Mac OS, Linux.
Classification: 2.7, 4.9, 7.7.
Nature of problem:
The software computes 2D or 3D stationary solutions of the Gross–Pitaevskii equation with rotation. The main application is the computation of different types of vortex states (Abrikosov vortex lattice, giant vortex) in rotating Bose Einstein condensates. The software can be easily modified to take into account different related physical models.
Solution method:
The user has the choice between two robust and optimized numerical methods for the direct minimization of the Gross–Pitaevskii energy: a steepest descent method based on Sobolev gradients and a minimization

☆ This paper and its associated computer program are available via the Computer Physics Communication homepage on ScienceDirect (http://www.sciencedirect.com/science/journal/00104655).

* Corresponding author. Fax: +33 2 32 95 52 86.
E-mail addresses: vergez@ann.jussieu.fr (G. Vergez), ionut.danaila@univ-rouen.fr (I. Danaila), auliac@ann.jussieu.fr (S. Auliac), hecht@ann.jussieu.fr (F. Hecht).

http://dx.doi.org/10.1016/j.cpc.2016.07.034
0010-4655/© 2016 Elsevier B.V. All rights reserved.

algorithm based on the state-of-the-art optimization library Ipopt. For both methods, mesh adaptivity strategies are implemented to reduce the computational time and increase the local spatial accuracy when vortices are present.

Running time:

From minutes for 2D configurations to hours for 3D cases (on a personal laptop). Complex 3D cases (with hundreds of vortices) may require several days of computational time.

© 2016 Elsevier B.V. All rights reserved.

1. Introduction

The Bose–Einstein condensate (BEC) is an ideal system to study superfluidity at a macroscopic level: it is a highly controllable quantum system which admits a simple theoretical description using the Gross–Pitaevskii equation (GPE) [1]. A great deal of attention has been lately devoted to the development of accurate numerical schemes to solve different forms of the GPE, from the classical (stationary or time-dependent) GPE, to systems of coupled GPEs (*e.g.* for two-component or spinor BEC) and more recent formulations (*e.g.* with non-local interactions or fractional GPE). For recent reviews of numerical methods for GPE, see [2–6].

Among all these formulations, the stationary GPE is used either to numerically generate an initial condition for the simulation of real-time dynamics of BEC, or to directly investigate physical features of experimentally observed BEC. In the former case, the stationary (ground state) solution which is the global minimizer of the GP energy is sought, while in the latter case, capturing local minima of the GP energy could be of interest since they represent excited (or metastable) states observed in experimental BEC configurations. The most striking example of how numerical solutions of the stationary GP equation were used to investigate physics is the study of quantized vortices in rotating BEC. Since superfluidity in BEC is closely related to the nucleation of quantized vortices, this topic has focused the attention of physical and mathematical communities during the last two decades. Numerous experimental and theoretical studies were devoted to the investigation of three-dimensional properties of single (straight or bent) vortex lines, vortex rings or Abrikosov vortex lattices (for a review of such physical systems, see the dedicated volumes [7–10]). Numerical simulations of the stationary three-dimensional (3D) GPE proved as a valuable investigation tool for all these topics, revealing properties of quantized vortices difficult to observe experimentally, suggesting new configurations, or supporting new physical or mathematical theories (*e.g.* [11–15]; for a review, see [16]).

The difficulty in computing solutions of the stationary GP equation with rotation comes from the presence in a condensate of a large number of vortices, with large gradients of atomic density in the vortex cores. This explains the use in the literature of discretization methods with high order spatial accuracy: Fourier spectral [17–19], sixth-order finite differences [13–15], sine-spectral [20,21], Laguerre–Hermite pseudo-spectral [22], hybrid discontinuous Galerkin discretizations based on polynomials and plane waves [23], etc. Several software packages for the classical stationary GPE were deposited in the CPC Program Library. They use different numerical methods: iterative diagonalization method [24], optimal damping algorithm [25,26], Crank–Nicolson scheme [27], Newton-like method with an approximate line-search strategy [28,29], fully-explicit fourth-order Runge–Kutta scheme [30], semi-implicit backward Euler scheme [31], etc. The spatial discretization is generally based on spectral [25,28,31] or finite-difference [27,29,30,26] methods. Provided programs are written in Fortran [25,27], C [29,30] or Matlab [28,30,31,26].

Numerical methods based on standard finite elements are less represented in this field. Vortex states in rotating BEC were computed using finite elements with fixed meshes [32,33,20,34] or dynamically adapted meshes [35], but only for 2D configurations. To the best of our knowledge, no finite-element programs exist in the CPC Program Library for the GP equation with rotation. The purpose of this paper is thus to distribute a finite-element solver for computing steady solutions of the GPE with rotation, in both 2D and 3D settings. The code was built as a toolbox for FreeFem++ [36,37], which is a free software (under LGPL license) using a large variety of triangular finite elements (linear and quadratic Lagrangian elements, discontinuous P^1, Raviart–Thomas elements, etc.) to solve partial differential equations. FreeFem++ is an integrated product with its own high level programming language and a syntax close to mathematical formulations, making the implementation of numerical algorithms very easy. Among the features making FreeFem++ an easy-to-use and highly adaptive software we recall the advanced automatic mesh generator, mesh adaptation, problem description by its variational formulation, automatic interpolation of data, colour display on line, postscript printouts, etc. FreeFem++ community is continuously growing, with thousands of users all over the world.

The present FreeFem toolbox, called GPFEM, provides two efficient numerical methods for computing stationary states with vortices, with the following novelties:

(i) the steepest-descent algorithm based on Sobolev gradients suggested in [38] and tested for 2D configurations in [35] was improved by adding an optimized line-search algorithm for the descent step and extended for 3D configurations;

(ii) a novel minimization method for 2D and 3D configurations was implemented based on the state-of-the-art optimization library Ipopt [39] using the direct minimization interior point method;

(iii) the mesh adaptivity algorithm suggested in [35] for 2D configurations was extended in 3D and optimized by the use of anisotropic mesh adaptivity functions provided by **mshmet** [40] and **mmg3d** [41] softwares.

From the programming point of view, the toolbox presents the following advantages:

(iv) the switch from different finite elements (from linear P^1 to quadratic P^2 and high-order P^3 or P^4 finite elements) implies the modification of a single instruction (the definition of the finite-element space);

(v) the scripts are easy to adapt to different mathematical or physical settings (two different scalings are implemented);

(vi) a graphical interface allows to run predefined 2D or 3D examples.

The paper is organized as follows. In Section 2, we present different mathematical formulations of the GP equation and energy. Two different scalings are introduced. Numerical methods are presented in Section 3 and the important issue of setting the initial condition for the computation is described in Section 4. The details of the derivation of closed formulae for the Thomas–Fermi approximation (generally used as initial condition) is deferred to Appendix. The structure of the provided software is described in great detail in Section 5. Various test cases for computing 2D and 3D configurations with vortices are presented in Section 6. The optional user interface is also described in Section 6. The main features of the software and possible extensions are summarized in Section 7.

2. Mathematical model: the Gross–Pitaevskii energy

2.1. The Gross–Pitaevskii energy for the rotating condensate

We consider in this paper numerical methods for the direct minimization of the Gross–Pitaevskii energy. For a pure BEC of N atoms confined in a trapping potential $V_{\rm trap}(\mathbf{x})$ rotating with angular velocity $\boldsymbol{\Omega}$, the energy of the system in the rotating frame is described by the functional:

$$\mathcal{E}(\psi) = \int_{\mathbb{R}^3}\left(\frac{\hbar^2}{2m}|\nabla\psi|^2 + V_{\rm trap}\,|\psi|^2 \right.$$
$$\left. + \frac{1}{2}g|\psi|^4 - \psi^*\boldsymbol{\Omega}\cdot\mathcal{L}(\psi)\right)\,d\mathbf{x} \qquad (1)$$

where $\psi(\mathbf{x})$ is the classical field complex wave function, ψ^* denotes its complex conjugate, m is the atomic mass, $\hbar$ the reduced Planck constant and g the coupling constant

$$g = 4\pi\hbar^2 a_s/m, \quad \text{with } a_s \text{ the scattering length.} \qquad (2)$$

The angular momentum $\mathcal{L}$ can be expressed as

$$\mathcal{L}(\psi) = \mathbf{x}\times\mathbf{P}(\psi), \quad \text{with the impulse} \quad \mathbf{P}(\psi) = -i\hbar\nabla\psi. \qquad (3)$$

We consider in the following rotations along the z-axis (*i. e.* $\boldsymbol{\Omega} = \Omega\,\mathbf{k}$) and therefore only the z-component of the angular momentum appears in (1) for the rotation term:

$$\boldsymbol{\Omega}\cdot\mathcal{L}(\psi) = \Omega\mathcal{L}_z\psi = i\hbar\Omega\left(y\frac{\partial\psi}{\partial x} - x\frac{\partial\psi}{\partial y}\right) = i\hbar\Omega\, A^t\nabla\psi,$$
$$\text{with} \quad A^t = (y, -x, 0). \qquad (4)$$

As a consequence, the form of the Gross–Pitaevskii energy considered in this paper is:

$$\mathcal{E}(\psi) = \int_{\mathbb{R}^3}\left(\frac{\hbar^2}{2m}|\nabla\psi|^2 + V_{\rm trap}\,|\psi|^2 + \frac{1}{2}g|\psi|^4\right)\,d\mathbf{x} - \Omega L_z, \qquad (5)$$

with L_z the total angular momentum:

$$L_z = i\hbar\int_{\mathbb{R}^3}\psi^*A^t\nabla\psi\, d\mathbf{x} = i\frac{\hbar}{2}\int_{\mathbb{R}^3}\left(\psi^*A^t\nabla\psi - \psi A^t\nabla\psi^*\right)d\mathbf{x}$$
$$= \hbar\int_{\mathbb{R}^3}\Re\left(i\psi^*A^t\nabla\psi\right)d\mathbf{x}. \qquad (6)$$

$\Re$ denotes the real part. We compute here minimizers of the energy (5) with the constraint

$$\int_{\mathbb{R}^3}|\psi(\mathbf{x})|^2\ d\mathbf{x} = N, \qquad (7)$$

expressing the conservation of the number of atoms in the condensate. Among these minimizers, the ground state is defined as a global minimum, *i. e.* $\psi_g = \min_\psi\,\mathcal{E}(\psi)$. Local minimizers with energy larger than that of the ground state are called excited states or meta-stable states.

Using (6), the energy (5) can be written in the following form that will be useful in deriving numerical methods in the next section:

$$\mathcal{E}(\psi) = \int_{\mathbb{R}^3}\left(\frac{\hbar^2}{2m}\left|\nabla\psi + i\frac{m\Omega}{\hbar}A^t\psi\right|^2\right.$$
$$\left. + V_{\rm trap}^{\rm eff}\,|\psi|^2 + \frac{1}{2}g|\psi|^4\right)\,d\mathbf{x}, \qquad (8)$$

where the effective trapping potential is the original potential diminished by the centrifugal term:

$$V_{\rm trap}^{\rm eff} = V_{\rm trap} - \frac{1}{2}m\Omega^2r^2, \quad r^2 = x^2+y^2. \qquad (9)$$

Another useful form of the energy corresponds to the grand potential of the system:

$$\Xi = \mathcal{E}(\psi) - \mu N = \mathcal{E}(\psi) - \mu\int_{\mathbb{R}^3}|\psi|^2\,d\mathbf{x}, \qquad (10)$$

where $\mu\in\mathbb{R}$ is the chemical potential of the condensate, introduced as a Lagrange multiplier for the constraint (7). The Euler–Lagrange equation ($\delta\Xi = 0$) corresponding to (10) leads to the stationary (or time-independent) GP equation:

$$-\frac{\hbar^2}{2m}\nabla^2\psi + V_{\rm trap}\psi + g|\psi|^2\psi - i\hbar\Omega A^t\nabla\psi = \mu\psi. \qquad (11)$$

The ground state and excited states are therefore eigenfunctions of the nonlinear eigenvalue problem (11).

We also consider in this paper two-dimensional (2D) configurations corresponding to disk-shape (or pancake) condensates. The dimension reduction from 3D to 2D can be done by approximating the 3D wave function by a factorized ansatz $\psi(x,y,z) = \psi_{2D}(x,y)\psi_3(z)$. For the precise form of the ansatz, the reader is referred to review papers [42,43]. For a mathematical justification of the dimension reduction from 3D to 2D equations, see [4]. By integrating out the z-dependence, previous forms of energy and stationary GPE stand, with $\mathbb{R}^3$ replaced by $\mathbb{R}^2$, with the caution that the non-linear interaction constant g expressed by (2) for the 3D setting has to be replaced by its *reduced* form in 2D. This constant will be prescribed as an input parameter of the computation.

2.2. Scaling and trapping potential

We consider in the following the Gross–Pitaevskii model set on $\mathbb{R}^d$, with $d = 3$ or 2. Various forms of scaling are used in the literature [44–46]. To allow the switch between different scalings, we introduce a parameter ε and define a general length scale as:

$$x_s = \frac{a_{\rm ho}}{\sqrt{\varepsilon}}, \quad a_{\rm ho} = \sqrt{\frac{\hbar}{m\omega_\perp}}, \qquad (12)$$

where $a_{\rm ho}$ is the harmonic oscillator length defined with respect to a reference trapping frequency $\omega_\perp$. By setting $\tilde{\mathbf{x}} = \mathbf{x}/x_s$ and

$$u = \frac{\psi}{\sqrt{N}\,x_s^{-d/2}} = \varepsilon^{-d/4}\frac{\psi}{\sqrt{N}\,a_{\rm ho}^{-d/2}}, \qquad (13)$$

the dimensionless GP energy (per particle) becomes:

$$E(u) = \frac{\mathcal{E}(\psi)}{N\frac{\hbar^2}{m}a_{\rm ho}^{-2}} = \frac{\mathcal{E}(\psi)}{N\hbar\omega_\perp}$$
$$= \varepsilon\int_{\mathbb{R}^d}\left[\frac{1}{2}|\nabla u|^2 + C_{\rm trap}\,|u|^2\right.$$
$$\left. + \frac{1}{2}C_g|u|^4 - iC_\Omega\, u^*A^t\nabla u\right]\,d\tilde{\mathbf{x}}, \qquad (14)$$

where

$$C_{\rm trap}(\tilde{x},\tilde{y},\tilde{z}) = \frac{1}{\varepsilon^2}\tilde{V}(\tilde{x},\tilde{y},\tilde{z}),$$
$$\tilde{V}(\tilde{x},\tilde{y},\tilde{z}) = \frac{1}{m\omega_\perp^2x_s^2}V_{\rm trap}(x,y,z), \qquad (15)$$

$$C_g = \sqrt{\varepsilon}\,\beta, \qquad \beta = \frac{4\pi N a_s}{a_{\rm ho}} \quad \text{(in 3D)}, \\ \beta = \beta_{2D} \quad \text{(given in 2D)}, \tag{16}$$

$$C_\Omega = \frac{1}{\varepsilon}\left(\frac{\Omega}{\omega_\perp}\right). \tag{17}$$

From the conservation law (7) we obtain that the wave function u is now normalized to unity:

$$\|u\|_2^2 = \int_{\mathbb{R}^d} \left|u(\tilde{\mathbf{x}})\right|^2 \, d\tilde{\mathbf{x}} = 1. \tag{18}$$

The total angular momentum (6) is now scaled in units of $\hbar$:

$$\tilde{L}_z = \frac{L_z}{N\hbar} = i\int_{\mathbb{R}^d} u^* A^t \nabla u d\tilde{\mathbf{x}} = \int_{\mathbb{R}^d} \Re\left(iu^* A^t \nabla u\right) d\tilde{\mathbf{x}}. \tag{19}$$

In this non-dimensional setting, the energy (8) takes the form:

$$E(u) \\ = \varepsilon \int_{\mathbb{R}^d} \left[\frac{1}{2}\left|\nabla u + iC_\Omega\, A^t u\right|^2 + C_{\rm trap}^{\rm eff}\,|u|^2 + \frac{1}{2}C_g|u|^4\right] d\tilde{\mathbf{x}}, \tag{20}$$

and the grand potential (10) becomes:

$$\tilde{\Xi} = E(u) - \tilde{\mu}\int_{\mathbb{R}^d} |u|^2\, d\tilde{\mathbf{x}}, \quad \tilde{\mu} = \frac{\mu}{\hbar\omega_\perp}. \tag{21}$$

The non-dimensional effective trapping potential corresponding to (9) is defined as:

$$C_{\rm trap}^{\rm eff} = C_{\rm trap} - \frac{1}{2}C_\Omega^2\,\tilde{r}^2 = \frac{1}{\varepsilon^2}\left(\tilde{V}(\tilde{x},\tilde{y},\tilde{z}) - \frac{1}{2}\left(\frac{\Omega}{\omega_\perp}\right)^2 \tilde{r}^2\right) \\ = \frac{1}{\varepsilon^2}\,\tilde{V}^{\rm eff}(\tilde{x},\tilde{y},\tilde{z}). \tag{22}$$

Finally, the dimensionless form of the stationary GP equation (11) becomes with this scaling:

$$-\frac{1}{2}\nabla^2 u + C_{\rm trap} u + C_g|u|^2 u - iC_\Omega A^t \nabla u = \frac{1}{\varepsilon}\tilde{\mu}\, u, \tag{23}$$

For the trapping potential, we consider in the following a general *quadratic* + *quartic* form that allows to recover the expressions used in most of the theoretical and experimental studies of rotating BEC. Starting from the following physical form of the trapping potential (harmonic potential + detuned laser beam, see [47]):

$$V_{\rm trap}(x,y,z) = \frac{m}{2}\left(\omega_x^2 x^2 + \omega_y^2 y^2 + \omega_z^2 z^2\right) + U_2\left(\frac{r}{w_2}\right)^2 \\ + U_4\left(\frac{r}{w_4}\right)^4, \tag{24}$$

we obtain from (15) and (22) the dimensionless effective potential:

$$\tilde{V}^{\rm eff}(\tilde{x},\tilde{y},\tilde{z}) = \frac{1}{2}\left(a_x\tilde{x}^2 + a_y\tilde{y}^2 + a_z\tilde{z}^2 + a_4\tilde{r}^4\right). \tag{25}$$

The non-dimensional coefficients are:

$$\begin{cases} a_x = \left(\dfrac{\omega_x}{\omega_\perp}\right)^2 - \left(\dfrac{\Omega}{\omega_\perp}\right)^2 + 2\left(\dfrac{U_2}{m\omega_\perp^2 w_2^2}\right), \\ a_y = \left(\dfrac{\omega_y}{\omega_\perp}\right)^2 - \left(\dfrac{\Omega}{\omega_\perp}\right)^2 + 2\left(\dfrac{U_2}{m\omega_\perp^2 w_2^2}\right), \\ a_z = \left(\dfrac{\omega_z}{\omega_\perp}\right)^2, \\ a_4 = \dfrac{2}{\varepsilon}\left(\dfrac{U_4\, a_{\rm ho}^2}{m\omega_\perp^2 w_4^4}\right). \end{cases} \tag{26}$$

The classical scaling used in the physical literature is recovered for $\varepsilon = 1$. In some mathematical studies [46,48] it was convenient to define ε as:

$$\varepsilon = \left(\frac{a_{\rm ho}}{8\pi N a_s}\right)^{2/5}. \tag{27}$$

This second scaling, referred as the Aftalion–Rivière (AR) scaling, is particularly appropriate for the Thomas–Fermi (TF) regime characterized by strong interactions (the kinetic energy is negligible compared to the interaction energy). This regime is attained when $Na_s/a_{\rm ho} \gg 1$, which is typically the case in experiments (*e.g.* [49–51,47]). In this case, ε is a small parameter ($\varepsilon \approx 10^{-2}$ in experiments). As a consequence, we notice from (16) that $C_g = \frac{1}{2\varepsilon^2}$ and the GP energy (20) becomes:

$$E(u) = \int_{\mathbb{R}^d}\left[\frac{\varepsilon}{2}\left|\nabla u + iC_\Omega\, A^t u\right|^2 + \frac{1}{\varepsilon}\tilde{V}^{\rm eff}\,|u|^2 + \frac{1}{4\varepsilon}|u|^4\right] d\tilde{\mathbf{x}}, \tag{28}$$

which is indeed dominated by the trapping and interaction terms. The AR scaling was successfully used in numerical simulation of 2D [32] or 3D [13–15] BEC configurations with vortices.

3. Numerical methods: direct minimization of the GP energy

We present in this section two numerical methods to compute minimizers $u(\tilde{\mathbf{x}})$ of the non-dimensional GP energy (14) or (20), with the constraint (18). The problem is set on a bounded domain $\mathcal{D} \in \mathbb{R}^d$, and homogeneous Dirichlet boundary conditions $u = 0$ are imposed on $\partial\mathcal{D}$. The size of $\mathcal{D}$ will be estimated from the Thomas–Fermi approximation (see Appendix), in order to ensure that the condensate lies inside $\mathcal{D}$. The parameters of the minimization problem are the angular velocity C_Ω, the non-linear interaction constant C_g and the trapping potential $C_{\rm trap}(\tilde{\mathbf{x}})$. For the sake of simplicity, the tilde notation for non-dimensional variables will be dropped in the following.

3.1. A steepest descent method based on Sobolev gradients

The first method implemented in our toolbox is the steepest descent method using the Sobolev gradients suggested in [38,35]. The algorithm starts from an initial state $u_0(\mathbf{x})$ and iterates following

$$u_{n+1} = u_n - \alpha_n\, \mathcal{G}_n, \tag{29}$$

where $\mathcal{G}_n$ represents the gradient of the energy functional at step n and α_n the descent step. The idea introduced in [38] was to define a gradient related to the form (20) of the energy. A new Hilbert space, denoted by $H_A(\mathcal{D},\mathbb{C})$, was defined and equipped with the inner product:

$$\langle u, v\rangle_{H_A} = \int_{\mathcal{D}} \langle u, v\rangle + \langle\nabla_A u, \nabla_A v\rangle, \tag{30}$$

where $\nabla_A = \nabla + iC_\Omega A^t$ and $\langle u, v\rangle = uv^*$ denotes the complex inner product. It was proved in [38] that the norm arising from the metric $\|\cdot\|_{H_A}$ is equivalent to the standard Sobolev H^1 norm. Hence the completion of $C^1(\mathcal{D},\mathbb{C})$ with respect to this metric consists of all members of H^1. As a consequence, the Riesz representation theorem in the Hilbert space $H_A = H^1$ allows to define the Sobolev gradient $\nabla_{H_A}E(u)$ as the unique member of H^1 such that, $\forall v \in H^1(\mathcal{D},\mathbb{C})$:

$$E'(u)v = \Re\langle\nabla_{L^2}E(u), v\rangle_{L^2} = \Re\langle\nabla_{H^1}E(u), v\rangle_{H^1} \\ = \Re\langle\nabla_{H_A}E(u), v\rangle_{H_A}. \tag{31}$$

Since the L^2 gradient of the GP energy can be easily derived from (14):

$$\nabla_{L^2} E(u) = 2\varepsilon \left(-\frac{1}{2}\nabla^2 u + C_{\text{trap}} u + C_g |u|^2 u - iC_\Omega A^t \nabla u \right), \tag{32}$$

the relationship (31) allows to compute the H_A gradient. Before using this gradient in the descent method (29), it will be projected onto the tangent space of the constraint (18). An explicit projection formula is derived in [38]. This technique is an alternative of the usual approach that re-normalize the solution u_{n+1} after each descent step.

Compared to the descent method presented in [38,35], where a fixed value of the descent step was used, the present method introduces an efficient estimation of the optimal descent step. Since general purpose line-search methods (Brent, Armijo, etc.) proved to be very time consuming for this problem, we finally used the particular line-minimization analysis specific to the GP energy. The minimizer α_n of the real function:

$$J_n(\alpha) = E(u_n - \alpha\, \mathscr{g}_n), \quad \alpha > 0 \tag{33}$$

is a root of the third order polynomial:

$$J_n'(\alpha) = c_3\alpha^3 + c_2\alpha^2 + c_1\alpha + c_0, \tag{34}$$

with coefficients

$$c_3 = 2C_g \int_{\mathcal{D}} |\mathscr{g}_n|^4 , \tag{35}$$

$$c_2 = -6C_g \int_{\mathcal{D}} |\mathscr{g}_n|^2 \; \Re\left(\langle u_n, \mathscr{g}_n\rangle\right), \tag{36}$$

$$\begin{aligned} c_1 \; = \; & \int_{\mathcal{D}} |\nabla \mathscr{g}_n|^2 + 2C_{\text{trap}} |\mathscr{g}_n|^2 + 2C_g\, |u_n|^2\, |\mathscr{g}_n|^2 \\ & + 4C_g \Re\left(\langle u_n, \mathscr{g}_n\rangle\right)^2 - 2C_\Omega \Re\left(i\mathscr{g}_n^* A^t \nabla \mathscr{g}_n\right), \end{aligned} \tag{37}$$

$$\begin{aligned} c_0 \; = \; & -\int_{\mathcal{D}} \Re\left(\langle \nabla u_n, \nabla \mathscr{g}_n\rangle\right) + 2\Re\left(\langle u_n, \mathscr{g}_n\rangle\right) \left[C_{\text{trap}} + C_g\, |u_n|^2\right] \\ & - 2C_\Omega \Re\left(i\mathscr{g}_n^* A^t \nabla u_n\right). \end{aligned} \tag{38}$$

In FreeFem++, we can use the function **polycomplexsolve** (from **GSL** library) [52] to calculate the three roots of the polynomial $J_n'(\alpha)$ and then select the root realizing the minimum of the energy $J_n(\alpha)$.

The algorithm for the descent method can be easily identified in the FreeFem++ scripts, since appropriate macros were defined for the mathematical operators (inner product, norms, etc.). All variables are discretized using P^1 finite elements; the non-linear term is represented with P^4 finite elements in 2D and P^2 in 3D. The following steps were programmed, with a syntax very close to mathematical relationships:

1. Suppose that the solution u_n at iteration n was built. We compute $G = \nabla_{H_A} E(u_n)/(2\varepsilon)$, solution of the variational problem corresponding to (31) and (32):

   $$\forall v \in H_0^1(\mathcal{D}, \mathbb{C}),$$

   $$\begin{aligned} &\int_{\mathcal{D}} \left(1 + C_\Omega^2 (x^2 + y^2)\right) Gv + \nabla G \nabla v - 2iC_\Omega A^t \nabla G v \\ &\quad = \int_{\mathcal{D}} \frac{1}{2} \nabla u_n \nabla v + \left[C_{\text{trap}} u_n + C_g |u_n|^2 u_n - iC_\Omega A^t \nabla u_n\right] v. \end{aligned} \tag{39}$$

2. We compute the projection of G over the tangent space of the unitary norm constraint (see [38]):

   $$P_{u_n,H_A} G = G - \frac{\Re\left(\langle u_n, G\rangle_{L^2}\right)}{\Re\left(\langle u_n, v_{H_A}\rangle_{L^2}\right)} v_{H_A}, \tag{40}$$

   where v_{H_A} is solution of the variational problem:

   $$\langle v_{H_A}, v\rangle_{H_A} = \langle u_n, v\rangle_{L^2}, \; \forall v \in H_A. \tag{41}$$

3. We compute the optimal descent step:

   $$\chi_n = \min_{\chi > 0} E(u_n - \chi\, P_{u_n,H_A} G), \tag{42}$$

   by finding the roots of the third order polynomial (34) with coefficients (35)–(38) and choosing the one realizing the minimum of the line energy. Note that the factor (2ε) appearing in the expression of the gradient (32) was included in the expression of the optimal descent step $\chi = (2\varepsilon)\alpha$.
4. We build the solution u_{n+1} at iteration $n + 1$:

   $$u_{n+1} = u_n - \chi_n\, P_{u_n,H_A} G.$$

5. Finally, we compute the relative error $\delta E_{n+1} = \frac{E(u_{n+1}) - E(u_n)}{E(u_{n+1})}$ and call the mesh adaptivity algorithm suggested in [35] (see below). Convergence to the stationary state is achieved if $\delta E_{n+1} < \varepsilon_c = 10^{-9}$.

3.2. Mesh adaptation

FreeFem++ includes a powerful mesh adaptivity tool (function **adaptmesh**) using metric control algorithms suggested in [53–56]. The main idea is to define a metric based on the Hessian and use a Delaunay procedure to build a new mesh such that all the edges are close to the unit length with respect to this new metric. In the steepest descent algorithm, we call this function after building u_{n+1} in the step 4 of the previous algorithm. Since our convergence criterion is based on the relative change of energy of the solution (δE_{n+1}) we use the same indicator to trigger the mesh adaptive procedure following the next algorithm (see also [35]):

1. choose a sequence of decreasing values $\varepsilon^i \geq \varepsilon_c$, that represent threshold values for the mesh adaptivity;
2. set $i = 1$;
3. if δE_{n+1} is decreasing and $\varepsilon^{i+1} < \delta E_{n+1} < \varepsilon^i$ and $\delta E_n > \varepsilon_c$, call the mesh adaptivity procedure; the solution u is interpolated on the new mesh and normalized to satisfy the unitary norm constraint;
4. if δE_{n+1} is increasing, *i. e.* large variations of the energy appear (*e. g.* if new vortices enter the domain), reconsider the previous bounds by setting $i \to i - 1$;
5. if step 3 was performed $N_{ad} \geq 1$ times, increase i to $i + 1$. Limiting the number of mesh refinements for the same threshold, is necessary since, at step 2, the interpolation on the new refined mesh and the normalization of the solution could lead to an increase of the value of δE_{n+1}.

Fig. 1 illustrates the 2D mesh adaptivity procedure. It represents a test case where the initial field has an off-centred vortex and the final (converged) solution is expected to present a centred vortex (the details of the parameters for this case are given in Section 6). We plot in Fig. 1(a) the initial state, built with the Thomas–Fermi approximation. In Fig. 1(b) we plot the final solution corresponding to the converged field obtained with the Sobolev gradient method. A zoom in the vortex area is displayed in Fig. 1(a1) and (b1). Note that the mesh adaptivity procedure generated a denser mesh near the position of the vortex; the number of triangles was decreased near the border, where the solution is smoother. However, the derefinement of the mesh must be used with caution for high rotation rates, when new vortices can nucleate in the condensate near the boundary. It was shown in [35] that this procedure decreases the CPU time and the number of iterations when compared with computation on fixed refined meshes.

For 3D computations, FreeFem++ uses the function **mshmet** [40] to compute the metrics and the function **mmg3d** [41] to build the new mesh corresponding to this metric. In Fig. 2, we plot

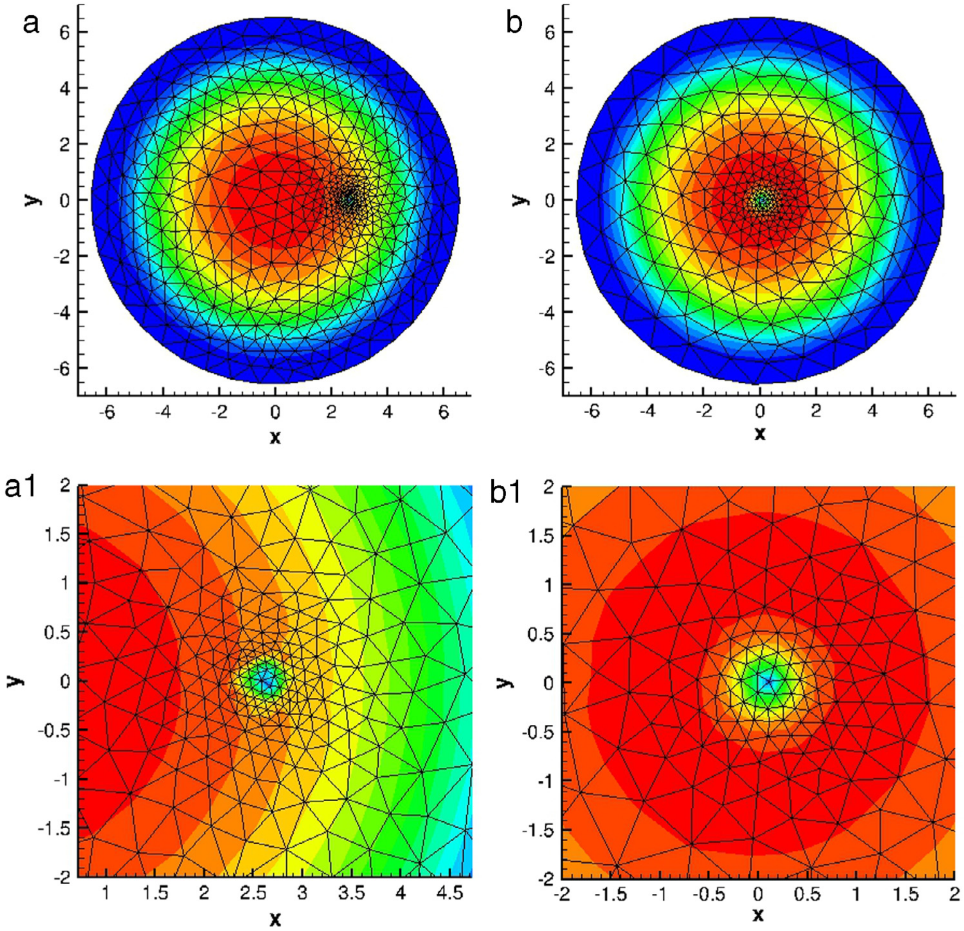


Fig. 1. Illustration of the mesh adaptivity in 2D. Test case with an initial state containing an off-centred vortex (a) and a final (stationary) state with a central vortex (b). The mesh refinement follows the evolution of the vortex position (corresponding zoom in figures a1 and b1).

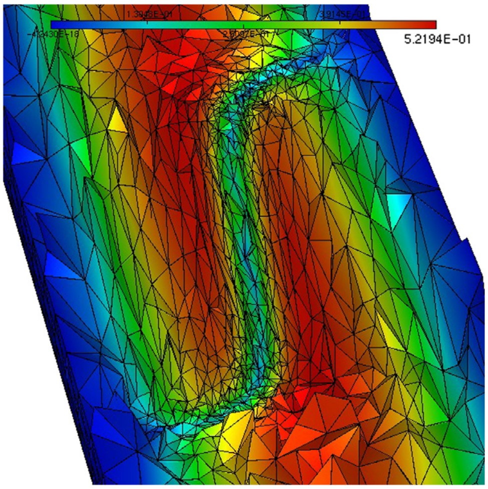


Fig. 2. Illustration of the mesh adaptivity in 3D. Test case computing the equilibrium configuration with a single S-shape vortex line.

a 3D mesh adapted to the solution presenting a vortex line with a “S” shape. We carried out the visualization with **medit**, a mesh visualization software [57] interfaced with FreeFem++. Note that the mesh adaptation follows precisely the vortex line by adding tetrahedra for a better accuracy. Outside the vortex area, the mesh adaptation allowed us to have fewer tetrahedra, with a bigger size.

3.3. Minimization algorithm using the optimization library Ipopt

The optimization library Ipopt is based on an interior point minimization method [39], a barrier functions tool [58] and a filter line search [59]. This powerful state-of-the-art optimization library is interfaced with FreeFem++ [60] and offers the possibility to solve constrained optimization problems of the general form:

$$\text{find } \mathbf{x_0} = \underset{\mathbf{x}\in\mathbb{R}^n}{\text{argmin}}(f(\mathbf{x})), \tag{43}$$

$$\text{such that } \begin{cases} \forall i \le n,\ x_i^{lb} \le x_i \le x_i^{ub} \text{ (simple bounds)}, \\ \forall i \le m,\ c_i^{lb} \le c_i(\mathbf{x}) \le c_i^{ub} \text{ (constraint functions)}, \end{cases} \tag{44}$$

where *lb* stands for *lower bound* and *ub* for *upper bound*. If for some $i \le m,\ c_i^{lb} = c_i^{ub}$ we obtain an equality constraint.

For the minimization of the Gross–Pitaevskii energy, the use of Ipopt is quite simple: the conservation constraint (18) is an equality constraint and, consequently, we take $m = 1$ and $c^{lb} = c^{ub} = 1$ in the previous general form. Ipopt will then solve the Euler–Lagrange equation associated to the problem (43)–(44):

$$\begin{cases} \nabla f(\mathbf{x}) + \lambda \nabla c(\mathbf{x}) = 0, \\ c(\mathbf{x}) = 0, \end{cases} \tag{45}$$

where $\lambda \in \mathbb{R}$ is a Lagrange multiplier and $c(\mathbf{x})$ the constraint. Note that, in our case, λ corresponds to the chemical potential. Let us define

$$L(\mathbf{x}, \lambda) := f(\mathbf{x}) + \lambda\, c(\mathbf{x}). \tag{46}$$

Ipopt first finds a descent direction $(\mathbf{dx},\ d\lambda)$ by using the Newton method. Indeed, at each iteration n it solves the system:

$$\begin{pmatrix} \nabla^2 L(\mathbf{x_n}, \lambda_n) & \nabla c(\mathbf{x_n}) \\ \nabla c(\mathbf{x_n}) & 0 \end{pmatrix} \begin{pmatrix} \mathbf{dx} \\ d\lambda \end{pmatrix} = - \begin{pmatrix} \nabla L(\mathbf{x_n}, \lambda_n) \\ c(\mathbf{x_n}) \end{pmatrix}. \tag{47}$$

Then it advances at the next step:

$$\begin{pmatrix} \mathbf{x_{n+1}} \\ \lambda_{n+1} \end{pmatrix} = \begin{pmatrix} \mathbf{x_n} \\ \lambda_n \end{pmatrix} + \alpha_n \begin{pmatrix} \mathbf{dx} \\ d\lambda \end{pmatrix},$$

where $\alpha_n \in (0, 1]$ is a descent step computed using the filter line-search method suggested in [59]. The algorithm will stop when either the error $(\varepsilon_n = \max\left(\|\nabla f(\mathbf{x_n}) + \lambda_n \nabla c(\mathbf{x_n})\|_\infty,\ \|c(\mathbf{x_n})\|_\infty\right))$ or the number of iterations reaches a value defined by the user.

As Ipopt seeks for solutions in $\mathbb{R}^n$, we have to separate in the Gross–Pitaevskii energy functional the real and imaginary part. The problem to solve becomes:

find $[u_r, u_i] \in (H_0^1(\mathcal{D}, \mathbb{R}))^2$ which minimizes

$$E(u_r, u_i) = \int_{\mathcal{D}} \left[\frac{1}{2} |\nabla u_r|^2 + \frac{1}{2} |\nabla u_i|^2 + C_{trap}\,(u_r^2 + u_i^2) + \frac{1}{2} C_g\,(u_r^2 + u_i^2)^2 \right] - C_\Omega\, L_z(u_r, u_i), \tag{48}$$

with

$$L_z(u_r, u_i) = \int_{\mathcal{D}} \left[y \left(\frac{\partial u_r}{\partial x} u_i - \frac{\partial u_i}{\partial x} u_r \right) - x \left(\frac{\partial u_r}{\partial y} u_i - \frac{\partial u_i}{\partial y} u_r \right) \right]. \tag{49}$$

Then we can calculate the Fréchet derivative of E as:

$$\begin{aligned} E'(u_r, u_i) \cdot [v_r, v_i] &= \int_{\mathcal{D}} \left[\nabla u_r \cdot \nabla v_r + \nabla u_i \cdot \nabla v_i + 2\, C_{trap}\,(u_r\, v_r + u_i\, v_i) \right] \\ &\quad + 2\, C_g \int_{\mathcal{D}} (u_r^2 + u_i^2)\,(u_r\, v_r + u_i\, v_i) \\ &\quad - C_\Omega L_z'(u_r, u_i) \cdot [v_r, v_i], \end{aligned} \tag{50}$$

with

$$\begin{aligned} -L_z'(u_r, u_i) \cdot [v_r, v_i] &= \int_{\mathcal{D}} y \left[-\frac{\partial u_r}{\partial x} v_i + \frac{\partial u_i}{\partial x} v_r - \frac{\partial v_r}{\partial x} u_i + \frac{\partial v_i}{\partial x} u_r \right] \\ &\quad + \int_{\mathcal{D}} x \left[\frac{\partial u_r}{\partial y} v_i - \frac{\partial u_i}{\partial y} v_r + \frac{\partial v_r}{\partial y} u_i - \frac{\partial v_i}{\partial y} u_r \right]. \end{aligned} \tag{51}$$

Finally, the second order Fréchet derivative of E is expressed as:

$$\begin{aligned} E''(u_r, u_i) \cdot ([v_r, v_i], [w_r, w_i]) &= \int_{\mathcal{D}} \left[\nabla v_r \cdot \nabla w_r + \nabla v_i \cdot \nabla w_i + 2\, C_{trap}\,(v_r\, w_r + v_i\, w_i) \right] \\ &\quad + 2\, C_g \int_{\mathcal{D}} \left[(u_r^2 + u_i^2)\,(v_r\, w_r + v_i\, w_i) \right] \\ &\quad + 4\, C_g \int_{\mathcal{D}} \left[(u_r\, v_r + u_i\, v_i)\,(u_r\, w_r + u_i\, w_i) \right] \\ &\quad - C_\Omega L_z''(u_r, u_i) \cdot ([v_r, v_i], [w_r, w_i]), \end{aligned} \tag{52}$$

with

$$\begin{aligned} -L_z''(u_r, u_i) \cdot ([v_r, v_i], [w_r, w_i]) &= \int_{\mathcal{D}} y \left[-\frac{\partial w_r}{\partial x} v_i + \frac{\partial w_i}{\partial x} v_r - \frac{\partial v_r}{\partial x} w_i + \frac{\partial v_i}{\partial x} w_r \right] \\ &\quad + \int_{\mathcal{D}} x \left[\frac{\partial w_r}{\partial y} v_i - \frac{\partial w_i}{\partial y} v_r + \frac{\partial v_r}{\partial y} w_i - \frac{\partial v_i}{\partial y} w_r \right]. \end{aligned} \tag{53}$$

The expression of the constraint functional is:

$$c(u_r, u_i) = \int_{\mathcal{D}} (u_r^2 + v_r^2) - 1, \tag{54}$$

and its gradient:

$$\nabla c(u_r, u_i) \cdot [v_r, v_i] = 2 \int_{\mathcal{D}} (u_r\, v_r + u_i\, v_i). \tag{55}$$

With Ipopt linked as an external library to FreeFem++, we cannot directly use mesh adaptivity in its internal algorithm. In exchange, we can couple the computation of the minimizer with the mesh adaptivity procedure. The following algorithm was implemented in the programs. Set n_{adapt}, the total number of mesh refinements to be done and ε_0 and $\varepsilon_{\text{last}}$, the first and the last mesh adaptivity prescribed errors (parameters of the FreeFem++ function **adaptmesh**).

1. At step $k \in [0, n_{\text{adapt}} - 1]$, run Ipopt to find a solution $[u_r^k, u_i^k]$.
2. Build a new mesh adapted to $[u_r^k, u_i^k]$ with a prescribed mesh adaptivity error

$$\varepsilon_k = \varepsilon_0 \left(\frac{\varepsilon_{\text{last}}}{\varepsilon_0} \right)^{k/(n_{\text{adapt}}-1)}. \tag{56}$$

3. Go to step $k + 1$.

Typical values used for 2D computations are $n_{\text{adapt}} = 4$, $\varepsilon_0 = 0.1$ and $\varepsilon_{\text{last}} = 0.005$. For 3D cases, as the computation is more difficult, it is more convenient to use a higher number of mesh adaptations and a lower ratio $\varepsilon_{\text{last}}/\varepsilon_0$. Typical values are $n_{\text{adapt}} = 6$, $\varepsilon_0 = 0.01$ and $\varepsilon_{\text{last}} = 0.005$.

4. Building the initial approximation

In computing stationary states for rotating BEC, the initial approximation used to start the iterative methods is of crucial importance. It can not only affect the convergence speed, but also the topology of the stationary solution, especially when local minima (meta-stable) solutions are sought. We present in this section three methods to build initial states for the computation of stationary solutions: the Thomas–Fermi approximation, a rapid calculation of the ground state with Ipopt for simplified configurations (axisymmetric or non-rotating) and, finally, an ansatz for a manufactured initial state with vortices.

4.1. Analytical solution based on the Thomas–Fermi approximation

The Thomas–Fermi regime is characterized by strong interactions (the kinetic energy is negligible compared to the interaction energy). This regime is attained when $Na_s/a_{\text{ho}} \gg 1$. If the healing length $\xi = (8\pi a_s \rho)^{-1/2}$, with ρ the atomic density, is defined as the length for which the kinetic and interaction energies are comparable, in the Thomas–Fermi regime the characteristic length scales are larger than the healing length. We give below some typical values from experiments of BEC with vortices [49,61,62]:

$$\begin{array}{ccccccccc} a_s & \ll & 1/\rho^{1/3} & < & \xi & \ll & a_{\text{ho}} & \ll & R \\ 5\,[\text{nm}] & \ll & 0.2\,[\mu\text{m}] & < & 0.3\,[\mu\text{m}] & \ll & 1\,[\mu\text{m}] & \ll & 5\,[\mu\text{m}], \end{array}$$

where $1/\rho^{1/3}$ approximates the distance between atoms and R is the radius of the condensate.

The general form of the Thomas–Fermi approximation of the atomic density $(\rho = |u|^2)$ is obtained by neglecting the first term in the energy (20). The Euler–Lagrange equation of the corresponding grand potential (21) gives:

$$\rho_{\text{TF}} = \left(\frac{\tilde{\mu}/\varepsilon - C_{\text{trap}}^{\text{eff}}}{C_g} \right)_+ = \frac{1}{\varepsilon^2 C_g} \left(\varepsilon \frac{\mu}{\hbar\omega_\perp} - \tilde{V}^{\text{eff}} \right)_+. \tag{57}$$

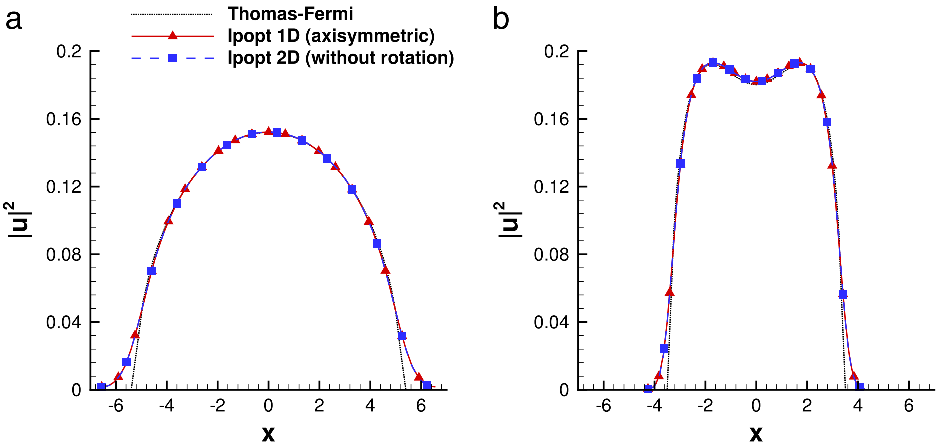


Fig. 3. Initialization of a 2D calculation. Density profiles corresponding to the Thomas–Fermi approximation (solid line), the axisymmetric solution computed with Ipopt (▲) and the full 2D solution computed with Ipopt without rotation (■). Harmonic potential (a) and quartic potential (b).

We notice that this form is equivalent to the usual Thomas–Fermi approximation for non-rotating condensates, but with a trapping potential (22) corrected by the centrifugal term (see also [63]). Following (9), for a harmonic trapping potential the radial trapping frequency $\omega_\perp$ is thus replaced by $(\omega_\perp^2 - \Omega^2)^{1/2}$.

It is also interesting to note from (13) and (57) that the atomic density in numerical simulations using the AR scaling with typical value $\varepsilon = 10^{-2}$ is amplified by a factor of 10^4, when compared to the classical scaling ($\varepsilon = 1$). This remark is important for setting the numerical value which will serve to identify a quantized vortex: since theoretically $\rho = 0$ in the vortex centre, the low value $\rho_{\min}$ of the iso-contour level used to represent vortices will depend on the scaling.

We use in the following the Thomas–Fermi approximation to estimate the size of the computational domain and also to set the initial guess for the minimization algorithms. We derive in the Appendix closed formulae for the Thomas Fermi approximation corresponding to different types of potentials: harmonic, *quartic + quadratic*, *quartic − quadratic*.

4.2. Numerical approximation with Ipopt for axisymmetric or non-rotating cases

The main drawback of the Thomas–Fermi approximation, which is generally a truncated parabola, is the discontinuity of its first derivative on the border of the condensate where $\rho_{\mathrm{TF}} = 0$. This could trigger oscillations of the solution, when high-order (spectral) methods are used for the space discretization. A smoother initial field can be obtained by directly computing with Ipopt a minimizer of the GP energy. When simplified forms of the energy (*e.g.* axisymmetric) are used, this preliminary computation is very cheap in terms of computational time.

We present below the approach of computing axisymmetric initial fields with Ipopt, corresponding to the ground state without vortices (a central vortex of given winding number can be easily added to the following expressions). We consider the cylindrical coordinates $(r,\ \theta,\ z)$ and assume that the solution is axisymmetric ($\frac{\partial u}{\partial \theta} = 0$) and symmetric in the z-direction ($u(z) = u(-z)$). This is also the case of the Thomas–Fermi approximation if $a_x = a_y$ in the trapping potential (25).

Using that $\frac{\partial u}{\partial \theta} = x\frac{\partial u}{\partial y} - y\frac{\partial u}{\partial x}$ and assuming that $L_z = 0$ (no central vortex), the energy becomes

$$E(u) = 4\pi \int_0^{R_{max}} \int_0^{z_{max}} \left[\frac{1}{2} \left(\left| \frac{\partial u}{\partial r} \right|^2 + \left| \frac{\partial u}{\partial z} \right|^2 \right) \right.$$

$$\left. + \ C_{\mathrm{trap}}^{\mathrm{eff}} |u|^2 + \frac{1}{2} C_g |u|^4 \right] rdr\, dz. \qquad (58)$$

The 3D problem is now reduced to a 2D problem. In order to solve this 2D problem with Ipopt, we need the Fréchet derivative of E and its Hessian:

$$E'(u).v = 4\pi \int_0^{R_{max}} \int_0^{z_{max}} \left[\frac{\partial u}{\partial r}\frac{\partial v}{\partial r} + \frac{\partial u}{\partial z}\frac{\partial v}{\partial z} \right.$$
$$\left. + \ 2C_{\mathrm{trap}}^{\mathrm{eff}} uv + 2C_g |u|^2 uv \right] rdr\, dz, \qquad (59)$$

$$E''(u)[v,\ w] = 4\pi \int_0^{R_{max}} \int_0^{z_{max}} \left[\frac{\partial v}{\partial r}\frac{\partial w}{\partial r} + \frac{\partial v}{\partial z}\frac{\partial w}{\partial z} + 2C_{\mathrm{trap}}^{\mathrm{eff}} vw \right.$$
$$\left. + \ 2C_g \left(vw|u|^2 + 2uv\Re(uw) \right) \right] rdrdz. \qquad (60)$$

In the case of a 2D simulation, the axisymmetry reduces the 2D problem to a 1D problem. In this case, the integration and the derivative with respect to z must be omitted in previous formulations. Fig. 3 shows a comparison between the Thomas–Fermi approximation and the axisymmetric solution computed with Ipopt for two trapping potentials (25): harmonic potential, with $a_x = a_y = 1$, $a_4 = 0$ and quartic potential, with $a_x = a_y = 1$, $a_4 = 0.5$. A third solution, obtained by using the full 2D formulation of the GP energy without rotation in Ipopt, is also plotted for reference. We notice the regularity of the axisymmetric solution in the vicinity of $\rho_{\mathrm{TF}} = 0$ and the good approximation it offers, when compared with the full 2D computation. For anisotropic potentials, we can still use the full (2D or 3D) formulation of the GP energy without rotation to compute with Ipopt an initial condition for the computations with rotation.

4.3. Manufactured initial state with vortices

Sometimes it is necessary to manufacture initial states by artificially including vortices. This could be useful when local minima, corresponding to meta-stable solutions, are sought. If $u(x, y, z)$ is the ground state without rotation (set by the TF approximation or computed with Ipopt), we can add vortices by multiplying u in each plane (x, y) by the following ansatz used in [14,15] for 3D simulations:

$$u_v(x, y) = \sqrt{\frac{1}{2}\left[1 + \tanh\left(\frac{4}{\varepsilon_v}(r_v - \varepsilon_v)\right)\right]} \cdot e^{i\theta_v}, \qquad (61)$$

where $r_v = \sqrt{(x-x_c)^2 + (y-y_c)^2}$ and $\theta_v = \operatorname{atan}\left(\frac{y-y_c}{x-x_c}\right)$ are the polar coordinates taken from the imposed centre (x_c, y_c) of the vortex and ε_v the vortex radius. In order to obtain a particular 3D shape of the vortex (U-shaped or S-shaped vortex, see Fig. 8), we can prescribe the position of the vortex centre in each transverse plane (x, y). For example, a S-vortex lying in the major (x, z) plane will have $y_c = 0$ and

$$\begin{cases} x_c(z) = -1 + \dfrac{\tanh\left[\alpha_v\left(1 + \frac{z}{\beta_v}\right)\right]}{\tanh(\alpha_v)}, & \text{if } z < 0, \\[2ex] x_c(z) = 1 + \dfrac{\tanh\left[\alpha_v\left(-1 + \frac{z}{\beta_v}\right)\right]}{\tanh(\alpha_v)}, & \text{if } z \geq 0, \end{cases} \tag{62}$$

where α_v and β_v respectively control the curvature and the length of the vortex.

5. Description of the programs

The methods described previously were implemented in separated 2D and 3D toolboxes based on the FreeFem++ software [37]. Using two input files, the toolbox offers to the user the choice between two scalings (classical or Aftalion–Rivière), three ways of computing the initial approximation (Thomas–Fermi, axisymmetric or non-rotating) and two methods to compute the ground state (Sobolev gradient or Ipopt). The main difference between the 2D and 3D codes is in the post-processing part: we can automatically count the number of vortices in 2D while it is more difficult in 3D. Also, the setting of input parameters is different: there are several additional parameters in 3D to control the shape of the vortex ansatz (I-shaped, S-shaped or U-shaped) and the shape of the initial mesh (cylindrical or ellipsoidal). Moreover, the user can choose to plot the evolution of the energy with **Gnuplot** [64] during the computation and the evolution of the solution with either the FreeFem++ plotting tool or using **Medit** [57]. In this section we first describe the architecture of the programs and the organization of the files. Then we focus on the list of input parameters and the structure of output files.

5.1. Program architecture

Fig. 4 gives a schematic overview of the content of the 3D toolbox. The 2D toolbox has similar architecture.

All files are provided in a directory called *BEC_$\mathcal{X}$D_ToolBox_FreeFem* where $\mathcal{X}$ is the dimension 2 or 3. This directory is organized as follows:

1. The *BEC_$\mathcal{X}$D_ToolBox.edp* is the main script.
2. The *Input* directory contains two files allowing the user to choose parameters:
   - *BEC_$\mathcal{X}$D_physic_param.dat* contains the parameters describing the physical case.
   - *BEC_$\mathcal{X}$D_run_param.dat* contains choices for the run.
3. The *Include* directory contains 9 files:
   - *BEC_$\mathcal{X}$D_Macros.idp* contains all the useful macros and functions.
   - *BEC_$\mathcal{X}$D_comput_param.idp* reads the parameters files and builds constants.
   - *BEC_$\mathcal{X}$D_comput_init_condition.idp* computes an initial approximation using either Thomas–Fermi or Ipopt.
   - *BEC_$\mathcal{X}$D_Ipoptaxi_init.idp* contains the script to be used with Ipopt axisymmetric in dimension ($\mathcal{X} - 1$) for the initial condition.
   - *BEC_$\mathcal{X}$D_Ipopt_init.idp* contains the script to be used with Ipopt in dimension $\mathcal{X}$ without rotation to build the initial condition.
   - *BEC_$\mathcal{X}$D_GradS_method.idp* solves the main problem with Sobolev gradient method.
   - *BEC_$\mathcal{X}$D_Ipopt_method.idp* solves the main problem with Ipopt method.
   - *BEC_$\mathcal{X}$D_plot_energ.idp* builds a Gnuplot script and runs Gnuplot in order to plot the energy and other relevant quantities.
   - *BEC_2D_results.idp* finds the number of vortices and gives their positions in 2D.
4. The *Examples* directory. In 2D, this directory contains 8 examples of input files allowing the user to choose between two cases of scaling, potential or method. To do so, in a terminal window the user can write, for example, the command line:

   FreeFem++ *BEC_$\mathcal{X}$D_ToolBox.edp*

   -run *Examples/GradS_Harm_run_param.dat*

   -param *Examples/AR_Harm_physic_param.dat*.

   This will run the program with an harmonic potential, the Aftalion–Rivière scaling and Sobolev gradient method. In 3D, this directory contains 6 files to run examples to compute a S-shaped or a U-shaped vortex, using either the Sobolev gradient method or Ipopt. A more precise description of these examples is provided in Section 6.
5. *GLUT* directory contains a C++ script that must be compiled to create a user interface with GLUT.
6. A makefile to compile the source code for the interface and a README file are also provided.

5.2. Input parameters

We focus now on the description of the input parameters. These are distributed in two files. In both files, comments are preceded by the usual // symbol and key words by the @ symbol. If the user wants to set a parameter, he has to enter its value after the corresponding key word. If a key word is not present in a file, a default value is given to the corresponding parameter. Some parameters must be specified by the user, otherwise the computation stops (see below). By default, the user has to use the two files provided in the *Input* directory. However, any input file can be used by entering the following command in a terminal window:

FreeFem++ *BEC_$\mathcal{X}$D_ToolBox.edp* -param *name_physics* -run *name_run*.

Here, *name_physics* is the name of the input file containing the physical parameters and *name_run* is the name of the input file containing the parameters for the computation.

(1) The first file in the *Input* directory, *BEC_$\mathcal{X}$D_physic_param.dat*, contains the physical parameters:

- @scaling, a string that can take the values *AR* or *Classical* depending on which scaling is chosen. A value must be given to this parameter.
- @kind, a boolean that takes the value 0 if one wants to set constants already built from (16), (17) and (26), or the value 1 if one wants to set the corresponding physical parameters. A value must be given to this parameter.
- If the 0 value was chosen for @kind the following parameters must be set to a real value:

  @beta ($=\beta$), the coefficient in front of the non linear part of the equation (see (16)),

  @Omop ($=\frac{\Omega}{\omega_\perp}$), the coefficient of the angular momentum (see (17)),

  @ax, @ay, @az and @a4 are the coefficients in the potential V_{trap} (see (26)).

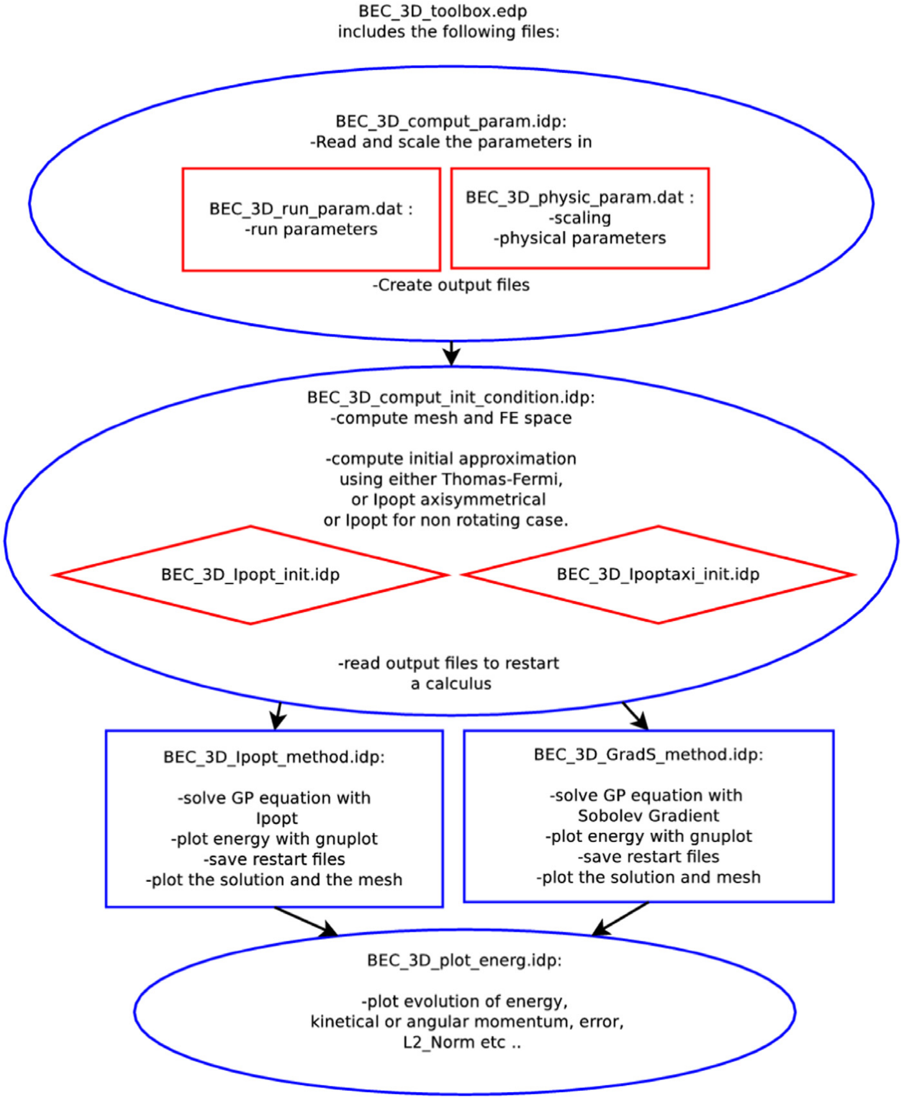

BEC_3D_toolbox.edp
includes the following files:
BEC_3D_comput_param.idp:
-Read and scale the parameters in
BEC_3D_run_param.dat :
-run parameters
BEC_3D_physic_param.dat :
-scaling
-physical parameters
-Create output files
BEC_3D_comput_init_condition.idp:
-compute mesh and FE space
-compute initial approximation
using either Thomas-Fermi,
or Ipopt axisymmetrical
or Ipopt for non rotating case.
BEC_3D_Ipopt_init.idp
BEC_3D_Ipoptaxi_init.idp
-read output files to restart
a calculus
BEC_3D_Ipopt_method.idp:
-solve GP equation with
Ipopt
-plot energy with gnuplot
-save restart files
-plot the solution and the mesh
BEC_3D_GradS_method.idp:
-solve GP equation with
Sobolev Gradient
-plot energy with gnuplot
-save restart files
-plot the solution and mesh
BEC_3D_plot_energ.idp:
-plot evolution of energy,
kinetical or angular momentum, error,
L2_Norm etc ..

Fig. 4. Program architecture of the 3D toolbox.

- If the value 1 was given to @kind the user has to assign a real value to the following parameters:

  @N, the number of atoms,

  @m, the atomic mass,

  @as, the scattering length,

  @Omega ($=\Omega$), the rotation speed,

  @omegax ($=\omega_x$), @omegay ($=\omega_y$), omegaz ($=\omega_z$), @omega2 ($=\omega_2$), @omega4 ($=\omega_4$),

  @U2 and @U4 are the coefficients in V_{trap} (see (24)).

(2) The second file, *BEC_$\mathcal{X}$D_run_param.dat* contains the parameters for the run:

1. Here are the parameters that must be set,
   - @method is a string to choose a method. The possible values are *Ipopt* or *GradS*.
   - @EPS0 is a real corresponding to the final error to reach.
   - @init is a string with the name of the initial approximation to use. The possible values are *TF* (Thomas–Fermi), Ipoptaxi (axisymmetric approximation) or Ipoptnorot (no rotation).

   All the parameters that follow are set by default:
   - @GradSMaxIter is the maximum number of iterations in the Sobolev gradient method. Default value: 8000.
   - @IpoptMaxIter is the maximum number of iterations between each mesh adaptation in Ipopt method. Default value: 50.
2. The following parameters are used for the outputs:
   - @dircase is a prefix of the name of the output directory. The form of potential and the name of the method used for computation are automatically added to this name. Default value: BEC_3D.
   - @scase is a prefix of the name of the output files. The values of $\frac{\Omega}{\omega_\perp}$ and C_g are automatically added to this name. Default value: BEC_3D.
   - @withplot is a boolean controlling the possibility of plotting the solution during the run. Default value: 1.
   - @savesol is a boolean controlling the possibility of saving the solution during the run. Default value: 1.
   - @IWAIT is a boolean controlling the possibility of waiting after each plot. Default value: 0.
   - @meditplot is a boolean controlling the possibility of plotting the solution with medit. Default value: 0.
   - @output is a string that takes the value *vtk* or *tecplot* for the outputs format. Default value: *tecplot*.
   - @ITERSAVE, @ITERNORM and @ITERPLOT are integers corresponding to the frequency of iterations in Sobolev gradient

method to save, normalize or plot the solution. Default value: 100.
- @savenergy and @plotenergy are booleans to save and plot the energy during the run. Default value: 1.
- @countvortices (only in 2D) is a boolean to count the number of vortices and to give their position. Default value: 1.

3. One can control how to build the initial mesh by setting the following parameters:
   - @aRdom is a coefficient that multiply the Thomas–Fermi radius in order to have a larger domain. Default value: 1.25.
   - @nbseg is the number of segments on the border of the mesh. Default value: 50 in 3D and 200 in 2D.
   - @meshkind (only in 3D) is a string that can take the values *cylindre* or *ellipsoid* and allows the user to choose between a cylindrical mesh or an ellipsoidal mesh in 3D. Default value: *ellipsoid*.
   - @hminsurf is the minimal size of the edge of a triangle on the surface of the ellipsoidal mesh. Default value: 0.6.
   - @hminvol is the minimal size of the edge of a tetrahedra inside the ellipsoidal mesh. Default value: 0.3.
4. The parameters for loading an old solution as an initial field are:
   - @ifILrst is a boolean, with true value if the user wants to load a restart file. Default value: 0. If ifILrst =1, the following 4 parameters have to be specified:
   - @keepmesh is a boolean to choose to keep the loaded mesh or not.
   - @dirload is a string for the name of the directory containing the restart mesh and solution.
   - @dmesh is a string for the name of the file containing the mesh to load.
   - @dsol is a string for the name of the file containing the solution to load.
5. The following parameters control how to build the initial field for the wave function:
   - @mod is an integer. If @mod $>$ 0, a central vortex with winding number @mod is added in the axisymmetric approximation built with Ipopt. Default value: 0.
   - @narray is the number of circles of vortices in the manufactured initial field (see Section 4.3). Default value: 0.

     If narray $=$ 1, the following 8 parameters have to be specified:
   - @Nv, the number of vortices on each circle.
   - @Rarr, the radius of the first circle.
   - @dRarr, the distance between two circles.
   - @Tharr, the orientation of the first circle.
   - @dTharr, a step between the orientation of each circle.
   - @shape (only in 3D) is a string controlling the shape of the ansatz vortex added in the initial condition (see Eq. (62)). It can be I, Ux, Uy, Sx, Sy. The x or y indicates if the vortex is in the plane (O, x, z) or (O, y, z). Default value: I.
   - @curvature ($=\alpha_v$, only in 3D) is the parameter controlling the curvature of the ansatz vortex (see (62)). Default value: 10.
   - @length ($=\beta_v$, only in 3D) is the parameter controlling the length of the ansatz vortex (see (62)). Default value: 2.
6. The user can control the mesh adaptivity process:
   - @ifladapt is a boolean to choose to adapt the mesh of the initial field. Default value: 1.
   - @erradaptI is the error in the mesh adaptation of the initial field. This parameter is used by the FreeFem function adaptmesh in 2D or mshmet in 3D. Default value: 0.01 in 3D and 0.1 in 2D.
   - @ifRadapt is a boolean to choose to adapt the mesh during the computation. Default value: 1.
   - @hminad is the minimal size of an edge in the new mesh. Default value: 0.001.
   - @hmaxad is the maximal size of an edge in the new mesh. Default value: 1.
   - @erradapt is the error in the mesh adaptation. It does change during a computation with Sobolev gradient method. If the Ipopt method is used for the computation, it corresponds to the parameter $\varepsilon_{\text{last}}$ in (56). Default value: 0.01 in 3D and 0.1 in 2D for Sobolev gradient method and 0.008 in 3D and 0.005 in 2D for Ipopt method.
   - @anisoadapt is a real value. If @anisoadapt $>$ 0, the mesh adaptation will be anisotropic and the ratio between the size of the smallest and the biggest edges of each triangle will be bounded by @anisoadapt. Default value: 10.
7. The following parameters are needed for the mesh adaptation in the Sobolev gradient method only:
   - @EPSAD1 is the first value of the L^2 relative error the user wants to reach to make a mesh adaptation ($=\varepsilon^1$ in 3.2). Default value: 1e$-$2.
   - @EPSADMIN is the last stage ($=\varepsilon_c$ in 3.2). Default value: 1e$-$9.
   - @IPASSAL is the number of times a mesh adaptation is performed before changing the value of EPSAD1 (N_{ad} in 3.2). Default value: 2.
   - @EPSADSTEP is a factor to change the value of EPSAD1. Default value: 2.
   - @ITERADAPT is the maximum number of iterations between two mesh adaptations. If *ITERADAPT* $=$ 0, we do not use this criterion. Default value: 0.
8. The last parameters are for the mesh adaptation in Ipopt method:
   - @niadapt is the number of times a mesh adaptation is performed with the same error ε_k (see (56)). Default value: 1.
   - @nbadapt is the total number of mesh adaptations made during the computation (n_{adapt} in (56)). Default value: 6 in 3D and 4 in 2D.
   - @maerr1 is the initial error in mesh adaptation (ε_0 in (56)). Default value: 0.01.

5.3. Output files

When a computation starts, the *Output* directory is created. It contains a directory, whose name includes the prefix (defined by the parameter @dircase), the form of potential and the chosen method. This directory will contain an *.echo* file with a summary of the main parameters, informations on the run, names of the output files, final energy and the CPU time. The *plot.gp* file will contain a Gnuplot script that the user can run to plot the evolution of the energy, the error, the angular momentum or the L^2 norm of the solution. The *.mesh* and *.rst* file contains the mesh and the solution respectively. They can be used as a restart field. Finally, the *.tec* or *.vtk* files contain the solution for a given iteration (defined by the parameters ITERPLOT) in the format *tecplot* or *vtk*.

6. Examples and user interface

To simplify the understanding of parameter files, some examples are provided in the directory *Examples*. A user interface was implemented using the **GLUT** library [65] to run these examples or to run the toolbox with predefined parameter files. In this section, we first present the example files and some results of computations. Then, we focus on the use of the GLUT user interface.

6.1. 2D computations

The examples for 2D computations use two forms of the trapping potential. For each case, the use of both scalings and

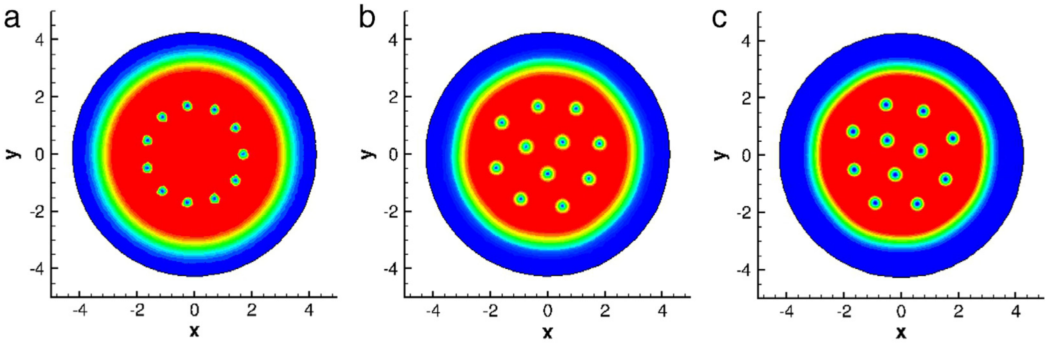


Fig. 5. Solution of the second example in 2D: (a) the initial state built by adding 11 manufactured vortices to the ground state computed with Ipopt (1D axisymmetric), (b) converged solution with the Sobolev gradient method, (c) solution obtained with Ipopt.

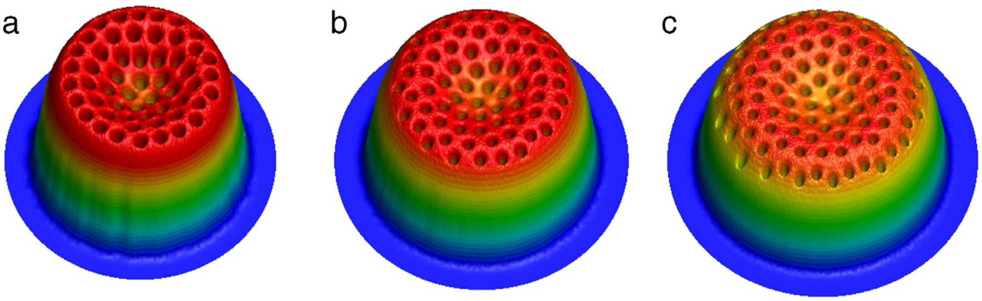


Fig. 6. 2D solution obtained with the Sobolev gradient method for a quartic + quadratic potential with $\Omega/\omega_\perp = 3.5$ and different values of the non-linear interaction constant: (a) $\beta = 5000$, (b) $\beta = 10\,000$, (c) $\beta = 15\,000$.

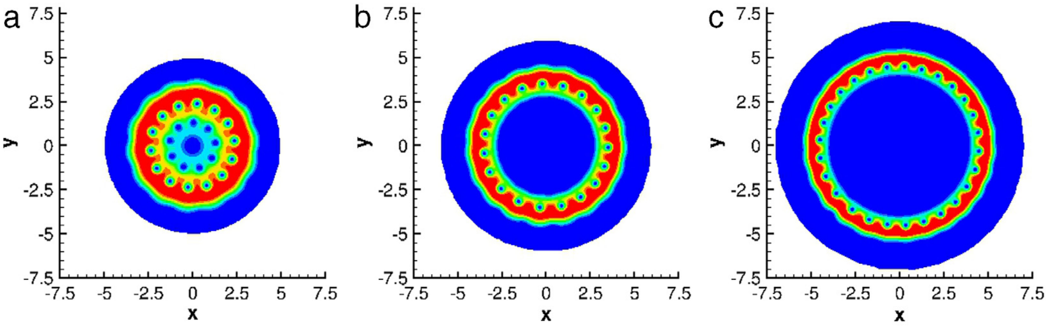


Fig. 7. 2D solution built with the Ipopt method for a quartic + quadratic potential with $\beta = 500$ and different values of the rotation frequency: (a) $\Omega/\omega_\perp = 3$, (b) $\Omega/\omega_\perp = 4$, (c) $\Omega/\omega_\perp = 5$.

numerical methods is possible.

1. The first case is the harmonic potential with $a_x = 1$, $a_y = 1$ (see (26)), $\beta = 500$ (see Eq. (16)) and $\Omega/\omega_\perp = 0.4$ (see Eq. (17)). We start with an initial approximation made with Ipopt axisymmetric and we add one manufactured off-centred vortex, as in Fig. 1(a). The final state we reach is a BEC with one central vortex as in Fig. 1(b). To run this example, the following files from the directory *Examples* have to be used:
   - *AR_Harm_physic_param.dat* or *Classical_Harm_physic_param.dat* for the physical parameters, depending on which scaling is chosen,
   - *Ipopt_Harm_run_param.dat* or *GradS_Harm_run_param.dat* for the computation parameters, depending on which method is chosen.
2. The second case is a combined quartic/quadratic potential with $a_x = 1$, $a_y = 1$, $a_4 = 0.5$ (see Eq. (26)), $\beta = 500$ and $\Omega/\omega_\perp = 2$. We start with an initial approximation made with Ipopt axisymmetric and we add a circle of manufactured vortices, as in Fig. 5(a). Both methods reach a BEC with eleven vortices organized into an Abrikosov lattice as shown in Fig. 5(b) and (c). To run this example, the following files must be selected by the user in the directory *Examples*:
   - *AR_Quart_physic_param.dat* or *Classical_Quart_physic_param.dat* for the physical parameters,
   - *Ipopt_Quart_run_param.dat* or *GradS_Quart_run_param.dat* for the chosen method.

In Figs. 6 and 7, we provide two results with the same quartic + quadratic potential as in the 2D example illustrated in Fig. 5: $a_x = 1$, $a_y = 1$, $a_4 = 0.5$. In the case illustrated in Fig. 6, we set the rotation speed to $\Omega = 3.5$ and increase the non-linear constant β from 5000 to 15 000. When this constant increases, the condensate becomes larger and the number of vortices increases significantly. They arrange in a triangular Abrikosov lattice. The files used to perform this simulation are provided in the directory *Input* as *BEC_2D_physic_param_Latt.dat* and *BEC_2D_run_param_Latt.dat*.

In the case of Fig. 7, the non-linear constant $\beta = 500$ is fixed and the rotation speed Ω increases from 3 to 5. The condensate is larger when the rotation speed increases and a giant vortex appears at the centre of the condensate. This case was simulated

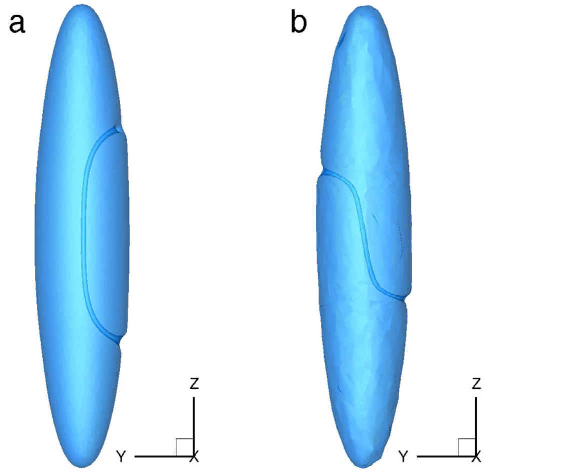


Fig. 8. Example from the interface software: isosurface of low atomic density illustrating 3D vortices. (a) U vortex obtained with the Ipopt method starting from a manufactured initial state with a U-shaped centred vortex. (b) S vortex obtained with the Sobolev gradient method starting from a manufactured initial state with a S-shaped centred vortex.

in [44]. The size of the computational domain increases as the rotation speed increases. This illustrates the need of the use of the Thomas–Fermi approximation to estimate the size of the domain. The files used to perform this simulation are provided in the directory *Input* under the names *BEC_2D_physic_param_Giant.dat* and *BEC_2D_run_param_Giant.dat*.

6.2. 3D computations

Two examples with harmonic trapping potential with $a_x = 1$, $a_y = 1.06^2$, $a_z = 0.067^2$ (see Eq. (26)), $\beta = 15\,900$ and $\Omega/\omega_\perp = 0.4$ are provided for 3D computations. They correspond to numerical tests used in [13]. These tests have shown that, with the same physical parameters, one can get different final meta-stable states, by starting from different initial states. In the first case, illustrated in Fig. 8(a), the computation starts with an axisymmetric approximation with a manufactured U-shaped vortex added at the centre. The final state, reached with both the Sobolev gradient method and the Ipopt method, presents a bended vortex with a U shape. In the second case of Fig. 8(b), we start with an axisymmetric approximation with a manufactured central vortex with a S shape. The final converged state keeps a S-shaped vortex when using both numerical methods. According to [13] the S-shaped vortices is a local minima of the energy. We conclude that both methods converge to the local minimum which is the closest to the initial guess provided. The input files used for these examples are provided in the directory *Examples* as:

- *VortexU_physic_param.dat* and *VortexS_physic_param.dat* for the physical parameters,
- *VortexU_GradS_run_param.dat* and *VortexU_Ipopt_run_param.dat* for the computation parameters of the U-shaped vortex case,
- *VortexS_GradS_run_param.dat* and *VortexS_Ipopt_run_param.dat* for the computation parameters of the S-shaped vortex case.

The result shown in Fig. 9 was obtained using physical parameters from [14]: $a_x = -0.2$, $a_y = -0.2$, $a_z = 0.067^2$, $a_4 = 0.075$, $\beta = 21\,000$ (see (26)), and $\Omega/\omega_\perp = 2$. The ground state displays a giant vortex surrounded by eleven singly-quantized vortices. This simulation was carried out using Ipopt for a quartic-minus-quadratic potential. The files used to perform this simulation are provided in the directory *Input* as: *BEC_3D_physic_param_Giant.dat* and *BEC_3D_run_param_Ipopt_Giant.dat*.

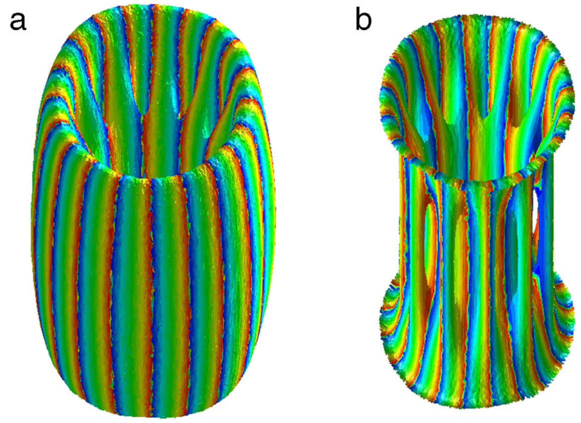


Fig. 9. 3D solution computed with the Ipopt method for a quartic-minus-quadratic potential. (a) Isosurface of low atomic density coloured with the phase. (b) The outer layer is removed to see the singly quantized vortices inside the condensate. $\beta = 21\,000$ and $\Omega/\omega_\perp = 2$.

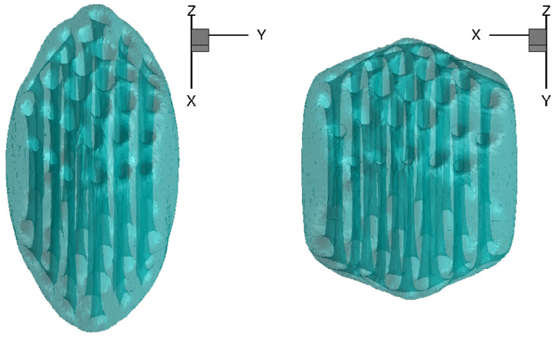
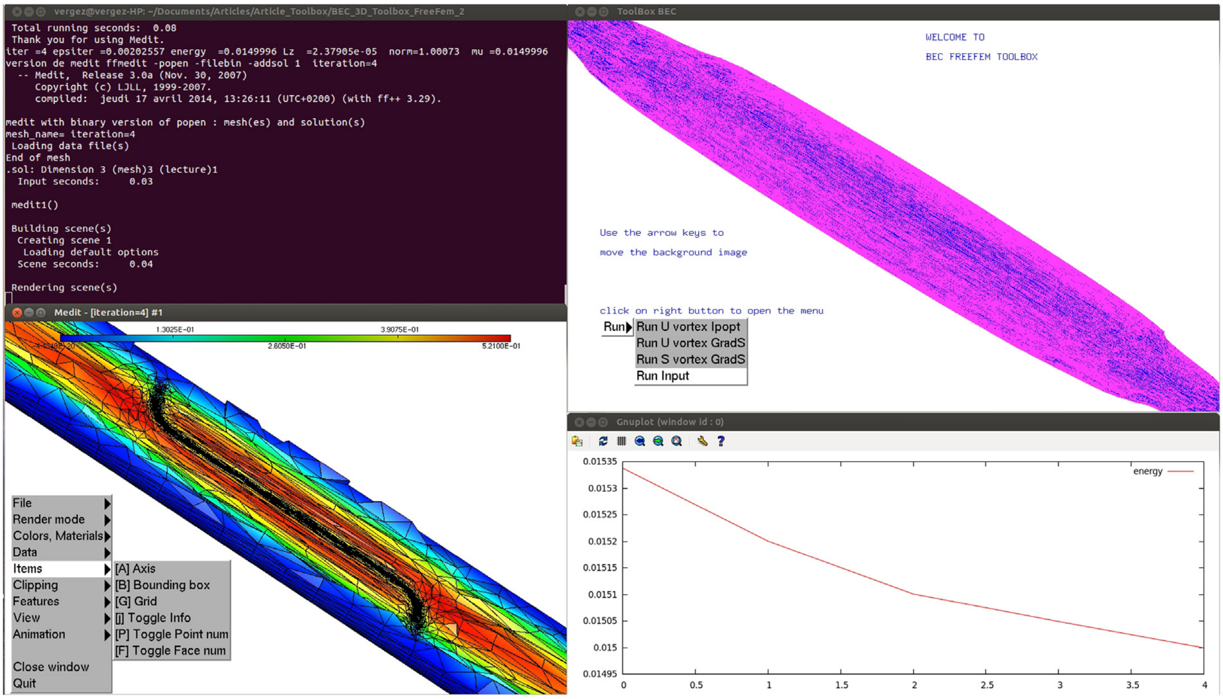


Fig. 10. 3D solution computed with the Sobolev gradient method for an anisotropic harmonic potential. Different views of an isosurface of the low atomic density showing the presence of 31 singly quantized vortices in an elongated condensate. $\beta = 50\,000$ and $\Omega/\omega_\perp = 0.95$.

Fig. 10 illustrates other possible vortex states that can be obtained. An anisotropic harmonic potential with $a_x = 1$, $a_y = 1.06^2$, $a_z = 0.067^2$, $\beta = 50\,000$ (see (26)), and $\Omega/\omega_\perp = 0.95$ was used. The computation resulted in an Abrikosov lattice with 31 vortices in 3D. The anisotropy makes the condensate to take an elongated shape following the *x*-axis. The files used to perform this simulation are provided in the directory *Input* as: *BEC_3D_physic_param_aniso.dat* and *BEC_3D_run_param_Ipopt_aniso.dat*.

6.3. Optional user interface

A simple user interface was made in C++ with the GLUT tool of the OpenGL library. The C++ source code for this interface is in the directory GLUT. It can be compiled using the makefile provided with the toolbox. This interface allows the user to easily run the examples or to run the toolbox using any modified input file. The screen capture of the interface in 3D is shown in Fig. 11. On the top left corner one can see a terminal from which was run the executable "RunToolbox". The window on the top right corner of Fig. 11 appears. By clicking on the right button of the mouse, a pull-down menu allows the user to run the toolbox with one of the three example files provided, or using the input files from the *Input* directory. Then a Gnuplot window appears plotting the evolution of the energy during the run. This window is on the bottom right corner of Fig. 11. Finally the bottom left corner of Fig. 11 shows the 3D solution plotted with Medit. The user can also decide to plot it with the usual graphical interface of FreeFem++.


vergez@vergez-HP: ~/Documents/Articles/Article_Toolbox/BEC_3D_Toolbox_FreeFem_2

```
 Total running seconds:  0.08
 Thank you for using Medit.
iter =4 epsiter =0.00202557 energy  =0.0149996 Lz  =2.37905e-05  norm=1.00073  mu =0.0149996
version de medit ffmedit -popen -filebin -addsol 1  iteration=4
  -- Medit,  Release 3.0a (Nov. 30, 2007)
     Copyright (c) LJLL, 1999-2007.
     compiled:  jeudi 17 avril 2014, 13:26:11 (UTC+0200) (with ff++ 3.29).

medit with binary version of popen : mesh(es) and solution(s)
mesh_name= iteration=4
 Loading data file(s)
End of mesh
.sol: Dimension 3 (mesh)3 (lecture)1
  Input seconds:     0.03

 medit1()

 Building scene(s)
  Creating scene 1
   Loading default options
  Scene seconds:     0.04

 Rendering scene(s)
```

WELCOME TO

BEC FREEFEM TOOLBOX

Use the arrow keys to

move the background image

click on right button to open the menu

Fig. 11. Screen capture of the user interface for a 3D computation. We can see: the terminal in which the application is run, the solution plotted with Medit and the plot of the energy evolution.

In 2D, there are three menus to choose one of the examples previously described:

1. *Potential* allows to choose between the harmonic (*Harm*) or the quartic + quadratic (*Quart*) trapping potential example.
2. *Method* allows to choose between the Ipopt (*Ipopt*) or the Sobolev gradient (*GradS*) method.
3. *Scaling* allows to choose between the Aftalion–Rivière (*AR*) or the classical scaling.

In the last menu (*Run*), the user can run either the selected example (*Run Example*) or run the toolbox with the input files which are in the *Input* directory (*Run Input*).

7. Conclusion

We provide with this paper a finite-element software for 2D and 3D computation of stationary solutions of the Gross–Pitaevskii equation. The user has the choice between two robust and optimized numerical methods: a steepest descent method based on Sobolev gradients and a minimization algorithm based on the state-of-the-art optimization library Ipopt. For both methods, mesh adaptivity strategies are implemented to reduce the computational time and increase the local spatial accuracy when vortices are present. The numerical system is tested and validated through various cases representing 2D and 3D configurations of Bose–Einstein condensates in rotation. A particular attention was paid to the physical interpretation of the computations. The main parameters of the run can be prescribed either in non-dimensional or physical form. Thomas–Fermi approximations are derived as closed formulae for a more accurate description of the initial field for the minimization procedures. Energy and angular momentum are tracked during the computation and post-processing tools allow to identify quantized vortices in the final, equilibrium state.

An optional graphical user interface is also provided with the software. It allows to easily run predefined cases or with user-defined parameter files.

The programs were written as a toolbox to be used within the free software FreeFem++. This offers the advantage that all technical issues related to the implementation of the finite element method are hidden, allowing to focus on numerical algorithms and their performance. Automatic mesh generators, powerful mesh adaptivity functions and the availability of various types of finite elements with complex functions are the main features making FreeFem++ very appealing in implementing numerical methods for Schrödinger type equations. The toolbox distributed with this paper is extremely versatile and can be easily adapted to deal with different physical models. A natural extension of this toolbox is the simulation of the time-dependent Gross–Pitaevskii equation: this is an ongoing work and will be reported in a further contribution.

Acknowledgements

This work was supported by the French ANR grant ANR-12-MONU-0007-01 BECASIM (Modéles Numériques call). We would like to acknowledge the use of computational resources provided by CRIHAN (Centre de Ressources Informatiques de Haute-Normandie, France) under the project 2015001. We also acknowledge the generous hospitality of the Fields Institute, Toronto, during the Thematic Program on Multiscale Scientific Computing, January–April, 2016.

Appendix. Formulae for the Thomas–Fermi approximation

We derive in this Appendix closed formulae for the Thomas–Fermi approximation for different types of trapping potentials (quartic ± quadratic). The Thomas–Fermi density (57) can be rewritten using (22) as:

$$\rho_{\mathrm{TF}} = |u|^2 = \left(\frac{\rho_0 - 2\tilde{V}^{\mathrm{eff}}}{C_S}\right)_+, \quad \rho_0 = 2\varepsilon\tilde{\mu} = 2\varepsilon\frac{\mu}{\hbar\omega_\perp}, \quad \text{and} \quad C_S = 2\varepsilon^2 C_g. \tag{A.1}$$

We recall that $\varepsilon = 1$ for the classical scaling using the oscillator length a_{ho} as length scale. The constant ρ_0 will be determined by imposing the unitary norm constraint (18). We derive below different formulas for ρ_0 corresponding to the effective trapping potential (25). We drop in the following the tilde notation.

A.1. 2D harmonic potential

For this case, the effective trapping potential (25) is reduced to

$$\tilde{V}^{\mathrm{eff}} = \frac{1}{2}\left(a_x x^2 + a_y y^2\right). \tag{A.2}$$

The unitary norm constraint (18) becomes

$$I = \int_{\mathcal{D}} \left(\rho_0 - a_x x^2 - a_y y^2\right)\, dxdy = C_S. \tag{A.3}$$

To calculate I analytically, we use the change of variables:

$$\begin{cases} x = \dfrac{r}{\sqrt{a_x}} \cos\theta \\ y = \dfrac{r}{\sqrt{a_y}} \sin\theta \end{cases} \quad dxdy = \frac{r}{\sqrt{a_x a_y}}\, drd\theta,$$
$$r \in [0, \sqrt{\rho_0}], \quad \theta \in [0, 2\pi], \tag{A.4}$$

and

$$I = \frac{1}{\sqrt{a_x a_y}} \int_0^{2\pi} d\theta \int_0^{\sqrt{\rho_0}} (\rho_0 - r^2) r dr = \frac{\pi \rho_0^2}{2\sqrt{a_x a_y}}. \tag{A.5}$$

Finally, the constant ρ_0 is expressed as:

$$\rho_0 = \left(\frac{2\sqrt{a_x a_y}}{\pi} C_S\right)^{1/2}, \tag{A.6}$$

and the dimensions of the condensate follow:

$$R_x = \sqrt{\frac{\rho_0}{a_x}}, \quad R_y = \sqrt{\frac{\rho_0}{a_y}}. \tag{A.7}$$

A.2. 3D harmonic potential

Same analysis for the potential

$$\tilde{V}^{\mathrm{eff}} = \frac{1}{2}\left(a_x x^2 + a_y y^2 + a_z z^2\right). \tag{A.8}$$

The constraint (18) becomes

$$I = \int_{\mathcal{D}} \left(\rho_0 - a_x x^2 - a_y y^2 - a_z z^2\right)\, dxdydz = C_S. \tag{A.9}$$

To calculate I analytically, we use the change of variables:

$$\begin{cases} x = \dfrac{r}{\sqrt{a_x}} \sin\theta\cos\phi \\ y = \dfrac{r}{\sqrt{a_y}} \sin\theta\sin\phi \\ z = \dfrac{r}{\sqrt{a_z}} \cos\theta \end{cases} \quad dxdydz$$
$$= \frac{r^2 \sin\theta}{\sqrt{a_x a_y a_z}}\, drd\theta d\phi, \quad \begin{cases} r \in [0, \sqrt{\rho_0}] \\ \theta \in [0, \pi] \\ \phi \in [0, 2\pi] \end{cases} \tag{A.10}$$

and

$$\begin{aligned} I &= \frac{1}{\sqrt{a_x a_y a_z}} \int_0^{2\pi} d\phi \int_0^{\pi} \sin\theta d\theta \int_0^{\sqrt{\rho_0}} (\rho_0 - r^2) r^2 dr \\ &= \frac{8\pi \rho_0^{5/2}}{15\sqrt{a_x a_y a_z}}. \end{aligned} \tag{A.11}$$

Finally, the constant ρ_0 is expressed as:

$$\rho_0 = \left(\frac{15\sqrt{a_x a_y a_z}}{8\pi} C_S\right)^{2/5}, \tag{A.12}$$

and the dimensions of the condensate follow:

$$R_x = \sqrt{\frac{\rho_0}{a_x}}, \quad R_y = \sqrt{\frac{\rho_0}{a_y}}, \quad R_z = \sqrt{\frac{\rho_0}{a_z}}. \tag{A.13}$$

A.3. 2D combined quartic and quadratic potential

We consider that the trap has radial symmetry ($a_x = a_y = a_2$) and the trapping potential is

$$\tilde{V}^{\mathrm{eff}} = \frac{1}{2}\left(a_2 r^2 + a_4 r^4\right). \tag{A.14}$$

Note that $a_4 > 0$, but a_2 can be either positive (quartic + quadratic potential) or negative (quartic − quadratic potential). The border of the condensate is defined by the radius R that satisfies:

$$a_4 R^4 + a_2 R^2 - \rho_0 = 0, \Longrightarrow R_{\pm}^2 = \frac{-a_2 \pm \sqrt{a_2^2 + 4\rho_0 a_4}}{2a_4}. \tag{A.15}$$

A.3.1. Case $a_2 \geq 0$: quartic + quadratic potential

In this case, $a_2 > 0$, $a_4 > 0$ and we infer from (A.15) that $\rho_0 > 0$ and there exists a single root R_+:

$$R_+^2 = \frac{-a_2 + \sqrt{a_2^2 + 4\rho_0 a_4}}{2a_4} > 0. \tag{A.16}$$

The constraint (18) becomes in polar coordinates (r, t):

$$I = 2\pi \int_0^R \left(\rho_0 - a_2 r^2 - a_4 r^4\right)\, rdr = C_S, \tag{A.17}$$

or

$$C_S = 2\pi \left[\rho_0 \frac{R^2}{2} - a_2 \frac{R^4}{4} - a_4 \frac{R^6}{6}\right] = \frac{\pi R^4}{6}\left(3a_2 + 4a_4 R^2\right). \tag{A.18}$$

Consequently, we first have to calculate the root $\eta = R^2 > 0$ of the non-linear equation (by a Newton method by example):

$$4a_4 \eta^3 + 3a_2 \eta^2 - \underbrace{\frac{6}{\pi}(C_S)}_{A_\eta} = 0, \tag{A.19}$$

and then calculate

$$\rho_0 = a_2 \eta + a_4 \eta^2 > 0. \tag{A.20}$$

The radius of the condensate is finally given by $R = \sqrt{\eta}$.

A.3.2. Case $a_2 < 0$: quartic − quadratic potential

We distinguish two cases:

- if $\rho_0 > 0$, (A.15) has only one root R_+, and the computation of ρ_0 is the same as above. We also infer that this case occurs when:

$$0 < \rho_0 = a_2 R^2 + a_4 R^4 \Longrightarrow \eta = R^2 > \frac{|a_2|}{a_4}. \tag{A.21}$$

- if $\rho_0 < 0$, (A.15) has two roots R_-, R_+ and the integration will be carried for $r \in [R_-, R_+]$ (there is a hole in the centre of the condensate):

$$I = 2\pi \int_{R_-}^{R_+} \left(\rho_0 - a_2 r^2 - a_4 r^4\right)\, rdr = C_S, \tag{A.22}$$

or

$$\frac{C_S}{2\pi} = \left[\frac{\rho_0}{2}\left(R_+^2 - R_-^2\right) - \frac{a_2}{4}\left(R_+^4 - R_-^4\right) - \frac{a_4}{6}\left(R_+^6 - R_-^6\right)\right] \qquad (A.23)$$

and using

$$R_+^2 + R_-^2 = -\frac{a_2}{a_4},\ R_+^2 R_-^2 = -\frac{\rho_0}{a_4},\ R_+^2 - R_-^2 = \frac{\sqrt{a_2^2 + 4\rho_0 a_4}}{a_4} \qquad (A.24)$$

we obtain that

$$\frac{C_S}{2\pi} = \frac{\left(a_2^2 + 4\rho_0 a_4\right)^{3/2}}{12a_4^2}, \qquad (A.25)$$

and finally

$$\rho_0 = \frac{1}{4a_4}\left[\left(\frac{6a_4^2}{\pi}C_S\right)^{2/3} - a_2^2\right] = \frac{1}{4a_4}\left[\left(a_4^2 A_\eta\right)^{2/3} - a_2^2\right]. \qquad (A.26)$$

Since $\rho_0 < 0$, this occurs if

$$a_4 < \sqrt{\frac{\pi|a_2|^3}{6C_S}} = \frac{|a_2|^{3/2}}{\sqrt{A_\eta}}. \qquad (A.27)$$

A.3.3. Summary for the 2D combined quartic and quadratic potential

$$\tilde{V}^{\mathrm{eff}} = \frac{1}{2}\left(a_2 r^2 + a_4 r^4\right). \qquad (A.28)$$

−− > compute ρ_0

$$\left[\begin{array}{l} \text{Compute}\quad C_S = 2\varepsilon^2 C_g, \\ \bullet\ \text{if } a_2 < 0 \text{ and } a_4 < \sqrt{\dfrac{\pi a_2^3}{6C_S}} \\ \rho_0 = \dfrac{1}{4a_4}\left[\left(\dfrac{6a_4^2}{\pi}C_S\right)^{2/3} - a_2^2\right] \\ \bullet\ \text{else} \\ \text{calculate the root } \eta > 0 \text{ of:} \\ f(\eta) = 4a_4\eta^3 + 3a_2\eta^2 - \dfrac{6}{\pi}\left(C_S\right) = 0, \\ \text{and then calculate} \\ \rho_0 = a_2 R^2 + a_4 R^4 \end{array}\right. \qquad (A.29)$$

−− > compute the maximum radius of the condensate

$$R_+ = \left(\frac{-a_2 + \sqrt{a_2^2 + 4\rho_0 a_4}}{2a_4}\right)^{1/2}.$$

A.4. 3D combined quartic and quadratic potential

We consider a trapping potential with radial symmetry ($a_x = a_y = a_2$):

$$\tilde{V}^{\mathrm{eff}} = \frac{1}{2}\left(a_2 r^2 + a_4 r^4 + a_z z^2\right). \qquad (A.30)$$

Note that $a_4 > 0, a_z > 0$, but a_2 can be either positive (quartic + quadratic potential) or negative (quartic−quadratic potential). The border of the condensate is defined by:

$$a_4 R^4 + a_2 R^2 + a_z z^2 - \rho_0 = 0,$$
$$\Longrightarrow z(r) = \pm\frac{1}{\sqrt{a_z}}\left(\rho_0 - a_2 r^2 - a_4 r^4\right)^{1/2}. \qquad (A.31)$$

A.4.1. Case $a_2 > 0$: quartic + quadratic potential

In this case, $a_2 > 0, a_z > 0, a_4 > 0$ and we infer from (A.31) that $\rho_0 > 0$ and it exists a single root $R_\perp$ which is the radius of the condensate in the central plane ($z = 0$):

$$R_\perp^2 = \frac{-a_2 + \sqrt{a_2^2 + 4\rho_0 a_4}}{2a_4} > 0. \qquad (A.32)$$

Consequently, the condensate extends in the central plane from $r = 0$ to $r = R_\perp$. Using the z-symmetry of the condensate, we calculate in cylindrical coordinates

$$I = \int_{\mathcal{D}} (\rho_0 - a_2 r^2 - a_4 r^4 - a_z z^2) = \int_0^{2\pi} d\theta \int_0^{R_\perp} rdr\, 2\int_0^{z(r)} (\rho_0 - a_2 r^2 - a_4 r^4 - a_z z^2) dz \qquad (A.33)$$

or using (A.31):

$$I = \frac{8\pi}{3\sqrt{a_z}} \int_0^{R_\perp} (\rho_0 - a_2 r^2 - a_4 r^4)^{3/2} rdr \qquad (A.34)$$

$$= \frac{8\pi}{3\sqrt{a_z}} \int_0^{R_\perp} \left[\left(\rho_0 + \frac{a_2^2}{4a_4}\right) - \left(\sqrt{a_4}r^2 + \frac{a_2}{\sqrt{4a_4}}\right)^2\right]^{3/2} rdr. \qquad (A.35)$$

It is useful to calculate the integral

$$J(x) = \int (\lambda^2 - x^2)\, dx, \quad \lambda > 0. \qquad (A.36)$$

After elementary integration by parts, we obtain:

$$J(x) = \frac{3\lambda^4}{8}\arcsin\left(\frac{x}{\lambda}\right) + \frac{3\lambda^2}{8}x\left(\lambda^2 - x^2\right)^{1/2} + \frac{1}{4}x\left(\lambda^2 - x^2\right)^{3/2}, \qquad (A.37)$$

or in the more useful form:

$$J(x) = \lambda^4\left[\frac{3}{8}\arcsin\left(\frac{x}{\lambda}\right) + \frac{3}{8}\left(\frac{x}{\lambda}\right)\left(1 - \left(\frac{x}{\lambda}\right)^2\right)^{1/2} + \frac{1}{4}\left(\frac{x}{\lambda}\right)\left(1 - \left(\frac{x}{\lambda}\right)^2\right)^{3/2}\right]. \qquad (A.38)$$

We also notice that:

$$J(\lambda) = \frac{3\pi}{16}\lambda^4, \quad J(0) = 0, \quad J(-\lambda) = -\frac{3\pi}{16}\lambda^4. \qquad (A.39)$$

Using now the notation

$$\lambda = \sqrt{\rho_0 + \frac{a_2^2}{4a_4}}, \qquad (A.40)$$

and the change of variables

$$u = \sqrt{a_4}r^2 + \frac{a_2}{\sqrt{4a_4}}, \quad du = 2\sqrt{a_4}dr, \qquad (A.41)$$

$$r = 0 \Longrightarrow u_0 = \frac{a_2}{\sqrt{4a_4}}, \qquad (A.42)$$

$$r = R_\perp \Longrightarrow u_\perp = \lambda, \qquad (A.43)$$

our integral becomes:

$$I = \frac{4\pi}{3\sqrt{a_z a_4}} \int_{u_0}^{\lambda} (\lambda^2 - u^2)^{3/2} du = \frac{4\pi}{3\sqrt{a_z a_4}} \left(J(\lambda) - J(u_0)\right). \quad \text{(A.44)}$$

Introducing the parameter:

$$\eta = \frac{a_2}{\sqrt{4a_4\rho_0}} > 0, \Longrightarrow \frac{u_0}{\lambda} = \frac{\eta}{\sqrt{1+\eta^2}}, \quad \text{(A.45)}$$

and using (A.38) and (A.39), we finally obtain:

$$\begin{aligned} I &= \frac{4\pi}{3\sqrt{a_z a_4}}\lambda^4 \left[\frac{3}{8}\left(\frac{\pi}{2} - \arcsin\frac{\eta}{\sqrt{1+\eta^2}}\right) \right. \\ &\quad \left. - \frac{3}{8}\frac{\eta}{\sqrt{1+\eta^2}}\frac{1}{(1+\eta^2)^{1/2}} - \frac{1}{4}\frac{\eta}{\sqrt{1+\eta^2}}\frac{1}{(1+\eta^2)^{3/2}}\right], \\ &= \frac{4\pi}{3\sqrt{a_z a_4}}\lambda^4 \left[\frac{3}{8}\arccos\frac{\eta}{\sqrt{1+\eta^2}} - \frac{3}{8}\frac{\eta}{1+\eta^2} \right. \\ &\quad \left. - \frac{1}{4}\frac{\eta}{(1+\eta^2)^2}\right]. \end{aligned} \quad \text{(A.46)}$$

Using that:

$$\lambda^4 = \rho_0^2(1+\eta^2)^2 = \frac{a_2^4}{(4a_4)^2}\frac{(1+\eta^2)^2}{\eta^4}, \quad \text{(A.47)}$$

we obtain a non-linear equation in η:

$$\begin{aligned} I = C_S &= \frac{8\pi a_2^4}{3a_z^{1/2}(4a_4)^{5/2}}\frac{1}{\eta^4}\left[\frac{3(1+\eta^2)^2}{8}\arccos\frac{\eta}{\sqrt{1+\eta^2}} \right. \\ &\quad \left. - \frac{3}{8}\eta(1+\eta^2) - \frac{1}{4}\eta\right], \\ &= \frac{\pi a_2^4}{a_z^{1/2}(4a_4)^{5/2}}\frac{1}{\eta^4}\left[(1+\eta^2)^2\arccos\frac{\eta}{\sqrt{1+\eta^2}} \right. \\ &\quad \left. - \eta^3 - \frac{5}{3}\eta\right]. \end{aligned} \quad \text{(A.48)}$$

To summarize this case, we have to

- find the root $\eta > 0$ of the non-linear equation:

$$\begin{aligned} f(\eta) &= \underbrace{\frac{a_z^{1/2}(4a_4)^{5/2}}{\pi a_2^4}C_S}_{A_\eta}\eta^4 - (1+\eta^2)^2\arccos\frac{\eta}{\sqrt{1+\eta^2}} \\ &\quad + \eta^3 + \frac{5}{3}\eta = 0, \end{aligned} \quad \text{(A.49)}$$

$$\begin{aligned} f'(\eta) &= 4A_\eta\eta^3 - 4\eta(1+\eta^2)\arccos\frac{\eta}{\sqrt{1+\eta^2}} \\ &\quad + (1+\eta^2) + 3\eta^2 + \frac{5}{3}, \end{aligned} \quad \text{(A.50)}$$

- compute

$$\rho_0 = \frac{a_2^2}{4a_4\eta^2}, \quad \text{(A.51)}$$

and the dimensions of the condensate:

$$R_\perp^2 = \frac{-a_2 + \sqrt{a_2^2 + 4\rho_0 a_4}}{2a_4}, \quad \text{(A.52)}$$

$$R_{zmax} = z|_{r=0} = \left(\frac{\rho_0}{a_z}\right)^{1/2}. \quad \text{(A.53)}$$

A.4.2. Case $a_2 = 0$: pure quartic potential

The integration is carried exactly in the same manner, the difference coming from the limits of the integration following r. We obtain

$$I = \frac{4\pi}{3\sqrt{a_z a_4}}\left(J(\lambda) - J(0)\right) = \frac{4\pi}{3\sqrt{a_z a_4}}\frac{3\pi}{16}\lambda^4, \quad \text{(A.54)}$$

with $\lambda = \sqrt{\rho_0}$. Finally

$$I = C_S = \frac{\pi^2}{2\sqrt{a_z 4a_4}}(\rho_0)^2, \quad \text{(A.55)}$$

and

$$\rho_0 = \frac{1}{\pi}\left(2a_z^{1/2}(4a_4)^{1/2}\right)^{1/2} C_S^{1/2}. \quad \text{(A.56)}$$

$$R_\perp = \left(\frac{\rho_0}{a_4}\right)^{1/4}, \quad R_{zmax} = \left(\frac{\rho_0}{a_z}\right)^{1/2}. \quad \text{(A.57)}$$

A.4.3. Case $a_2 < 0$: quartic − quadratic potential

For this case, $a_2 < 0$, $a_z > 0$, $a_4 > 0$ and we distinguish two subcases:

- If $\rho_0 < 0$, the condensate has a hole. We infer from (A.31) that there are two roots $R_\perp^\pm$

$$(R_\perp^\pm)^2 = \frac{-a_2 \pm \sqrt{a_2^2 + 4\rho_0 a_4}}{2a_4} > 0 \quad \text{(A.58)}$$

and the condensate extends in central plane from $R_\perp^-$ to $R_\perp^+$.

The integration is carried exactly in the same manner, the difference coming from the limits of the integration following r. We obtain

$$I = \frac{4\pi}{3\sqrt{a_z a_4}}\left(J(\lambda) - J(-\lambda)\right) = \frac{4\pi}{3\sqrt{a_z a_4}}2\frac{3\pi}{16}\lambda^4 \quad \text{(A.59)}$$

and finally

$$I = C_S = \frac{\pi^2}{\sqrt{a_z 4a_4}}\left(\rho_0 + \frac{a_2^2}{4a_4}\right)^2. \quad \text{(A.60)}$$

The value of ρ_0 results as:

$$\rho_0 = \frac{a_z^{1/4}(4a_4)^{1/4}}{\pi}C_S^{1/2} - \frac{a_2^2}{4a_4} = \frac{a_2^2}{4a_4}(\xi - 1). \quad \text{(A.61)}$$

Since $\rho_0 < 0$, this case is obtained if:

$$\xi = \frac{a_z^{1/4}(4a_4)^{5/4}}{\pi a_2^2}(C_S)^{1/2} = \frac{\sqrt{A_\eta}}{\sqrt{\pi}} < 1. \quad \text{(A.62)}$$

The dimensions of the condensate are

$$R_{max}^2 = \frac{-a_2 + \sqrt{a_2^2 + 4\rho_0 a_4}}{2a_4}, \quad \text{(A.63)}$$

$$R_{zmax} = z\Big|_{r^2 = \frac{-a_2}{2a_4}} = \frac{1}{\sqrt{a_z}}\sqrt{\rho_0 + \frac{a_2^2}{4a_4}}. \quad \text{(A.64)}$$

- If $\rho_0 > 0$, the condensate has only a depletion centred around $z = 0$ (the density profile has not any more the maximum at $z = 0$). This case occurs when:

$$\xi = \frac{a_z^{1/4}(4a_4)^{5/4}}{\pi a_2^2}C_S^{1/2} = \frac{\sqrt{A_\eta}}{\sqrt{\pi}} > 1. \quad \text{(A.65)}$$

The computation is the same as for the case of “quartic + quadratic” potential, with the difference that the root η is now negative. In particular

$$R_{zmax} = \frac{1}{\sqrt{a_z}}\sqrt{\rho_0 + \frac{a_2^2}{4a_4}}. \tag{A.66}$$

A.4.4. Summary for the 3D combined quartic and quadratic potential

$$\tilde{V}^{\text{eff}} = \frac{1}{2}\left(a_2 r^2 + a_4 r^4 + a_z z^2\right).$$

```
Compute   $C_S = 2\varepsilon^2 C_g$,
  • if $a_2 = 0$,
    $\rho_0 = \frac{1}{\pi}\left(2a_z^{1/2}(4a_4)^{1/2}\right)^{1/2} C_S^{1/2}$,
    $R_{max} = \left(\frac{\rho_0}{a_4}\right)^{1/4}$,   $R_{zmax} = \left(\frac{\rho_0}{a_z}\right)^{1/2}$.
  • else
  −− > compute
  $A_\eta = \frac{a_z^{1/2}(4a_4)^{5/2}}{\pi a_2^4} C_S$
  −− > define the function
  $f(\eta) = A_\eta\,\eta^4 - (1+\eta^2)^2 \arccos\frac{\eta}{\sqrt{1+\eta^2}} + \eta^3 + \frac{5}{3}\eta$
    • if $a_2 > 0$,
    −−−− > find the positive root $\eta \in [0, 200]$ of $f(\eta) = 0$
      $\rho_0 = \frac{a_2^2}{4a_4\eta^2}$,
      $R_{max} = \left(\frac{-a_2 + \sqrt{a_2^2 + 4\rho_0 a_4}}{2a_4}\right)^{1/2}$,   $R_{zmax} = \left(\frac{\rho_0}{a_z}\right)^{1/2}$.
    • else
    $\xi = \frac{\sqrt{A_\eta}}{\sqrt{\pi}}$,
      • if $\xi < 1$,
        $\rho_0 = \frac{a_2^2}{4a_4}(\xi - 1)$,
        $R_{max} = \left(\frac{-a_2 + \sqrt{a_2^2 + 4\rho_0 a_4}}{2a_4}\right)^{1/2}$,
        $R_{zmax} = \left(\frac{\rho_0}{a_z} + \frac{a_2^2}{4a_4 a_z}\right)^{1/2}$.
      • else
      find the **negative** root $\eta \in [-200, 0]$ of $f(\eta) = 0$
        $\rho_0 = \frac{a_2^2}{4a_4\eta^2}$,
        $R_{max} = \left(\frac{-a_2 + \sqrt{a_2^2 + 4\rho_0 a_4}}{2a_4}\right)^{1/2}$,
        $R_{zmax} = \left(\frac{\rho_0}{a_z} + \frac{a_2^2}{4a_4 a_z}\right)^{1/2}$.
```

References

[1] L.P. Pitaevskii, S. Stringari, Bose-Einstein Condensation, Clarendon Press, Oxford, 2003.

[2] A. Minguzzi, S. Succi, F. Toschi, M.P. Tosi, P. Vignolo, Phys. Rep. 395 (2004) 223–355.

[3] W. Bao, in: P.G. Kevrekidis, D.J. Frantzeskakis, R. Carretero-González (Eds.), Transport Phenomena and Kinetic Theory: Applications to Gases, Semiconductors, Photos, and Biological Systems, in: Series Modeling and Simulation in Science, Engineering and Technology, Birkhauser, 2006, pp. 215–255.

[4] W. Bao, Y. Cai, Kinet. Relat. Models 6 (2013) 1–135.

[5] X. Antoine, C. Besse, W. Bao, Comput. Phys. Comm. 184 (12) (2013) 2621–2633.

[6] W. Bao, Proceedings of the International Congress of Mathematicians (Seoul 2014) IV, 2014, pp. 971–996.

[7] C.F. Barenghi, R.J. Donnelly, W.F. Vinen (Eds.), Quantized Vortex Dynamics and Superfluid Turbulence, in: Lecture Notes in Physics, vol. 571, Springer, 2001.

[8] P.G. Kevrekidis, D.J. Frantzeskakis, R. Carretero-González (Eds.), Emergent Nonlinear Phenomena in Bose-Einstein Condensates, in: Atomic, Optical, and Plasma Physics, vol. 45, Springer, 2008.

[9] C.F. Barenghi, Y.A. Sergeev (Eds.), Vortices and Turbulence at Very Low Temperatures, in: CISM International Centre for Mechanical Sciences, vol. 501, Springer, 2008.

[10] B. Halperin, M. Tsubota (Eds.), Quantum Turbulence, in: Progress in Low Temperature Physics, vol. 16, Springer, 2009.

[11] K. Kasamatsu, M. Machida, N. Sasa, M. Tsubota, Phys. Rev. A 71 (2005) 063616.

[12] N.G. Berloff, Phys. Rev. A 69 (2004) 053601.

[13] A. Aftalion, I. Danaila, Phys. Rev. A 68 (2003) 023603(1–6).

[14] A. Aftalion, I. Danaila, Phys. Rev. A 69 (2004) 033608(1–6).

[15] I. Danaila, Phys. Rev. A 72 (2005) 013605(1–6).

[16] K. Kasamatsu, M. Tsubota, Prog. Low Temp. Phys. 16 (2008) 351–403.

[17] J.J. García-Ripoll, V.M. Pérez-García, Phys. Rev. A 64 (2001) 053611.

[18] J.J. García-Ripoll, V.M. Pérez-García, SIAM J. Sci. Comput. 23 (2001) 1315–1333.

[19] R. Zeng, Y. Zhang, Comput. Phys. Comm. 180 (2009) 854–860.

[20] W. Bao, Q. Du, SIAM J. Sci. Comput. 25 (2004) 1674.

[21] W. Bao, I.-L. Chern, F.Y. Lim, J. Comput. Phys. 219 (2006) 836–854.

[22] W. Bao, J. Shen, J. Comput. Phys. 227 (2008) 9778–9793.

[23] C. Farhat, J. Toivanen, J. Comput. Phys. 231 (2012) 4709–4722.

[24] R.P. Tiwari, A. Shukla, Comput. Phys. Comm. 174 (12) (2006) 966–982.

[25] C.M. Dion, E. Cancès, Comput. Phys. Comm. 177 (2007) 787–798.

[26] U. Hohenester, Comput. Phys. Comm. 185 (1) (2014) 194–216.

[27] P. Muruganandam, S. Adhikari, Comput. Phys. Comm. 180 (10) (2009) 1888–1912.

[28] M. Caliari, S. Rainer, Comput. Phys. Comm. 184 (3) (2013) 812–823.

[29] D. Vudragović, I. Vidanović, A. Balaz, P. Muruganandam, S.K. Adhikari, Comput. Phys. Comm. 183 (9) (2012) 2021–2025.

[30] R. Caplan, Comput. Phys. Comm. 184 (4) (2013) 1250–1271.

[31] X. Antoine, R. Duboscq, Comput. Phys. Comm. 185 (11) (2014) 2969–2991.

[32] A. Aftalion, Q. Du, Phys. Rev. A 64 (2001) 063603.

[33] W. Bao, W. Tang, J. Comput. Phys. 187 (2003) 230–254.

[34] L.O. Baksmaty, Y. Liub, U. Landmanc, N.P. Bigelowd, H. Pu, Math. Comput. Simulation 80 (2009) 131–138.

[35] I. Danaila, F. Hecht, J. Comput. Phys. 229 (2010) 6946–6960.

[36] F. Hecht, J. Numer. Math. 20 (2012) 251–266.

[37] F. Hecht, O. Pironneau, A.L. Hyaric, K. Ohtsuke, FreeFem++ (manual), 2007, www.freefem.org.

[38] I. Danaila, P. Kazemi, SIAM J. Sci. Comput. 32 (2010) 2447–2467.

[39] A. Wächter, An interior point algorithm for large-scale nonlinear optimization with applications in process engineering (Ph.D. thesis), Carnegie Mellon University, Pittsburgh, PA, USA.

[40] C.D.C. Dapogny, P. Frey, J. Comput. Phys. 262 (2014) 358–378.

[41] C. Dobrzynski, P. Frey, MMG3D: User Guide. [Technical Report] RT-0422, INRIA hal-00681813, 2012.

[42] F. Dalfovo, S. Giorgini, L.P. Pitaevskii, S. Stringari, Rev. Modern Phys. 71 (1999) 463–512.

[43] V.S. Bagnato, D.J. Frantzeskakis, P.G. Kevrekidis, B.A. Malomed, D. Mihalache, Romanian Rep. Phys. 67 (2015) 5–50.

[44] A.L. Fetter, B. Jackson, S. Stringari, Phys. Rev. A 71 (2005) 013605.

[45] M. Tsubota, K. Kasamatsu, M. Ueda, Phys. Rev. A 65 (2002) 023603.

[46] A. Aftalion, T. Rivière, Phys. Rev. A 64 (2001) 043611.

[47] V. Bretin, S. Stock, Y. Seurin, J. Dalibard, Phys. Rev. Lett. 92 (2004) 050403.

[48] A. Aftalion, Vortices in Bose-Einstein Condensates, Birkhauser, 2006.

[49] K.W. Madison, F. Chevy, W. Wohlleben, J. Dalibard, J. Modern Opt. 47 (2000) 2715.

[50] K.W. Madison, F. Chevy, V. Bretin, J. Dalibard, Phys. Rev. Lett. 86 (2001) 4443.

[51] P. Rosenbusch, V. Bretin, J. Dalibard, Phys. Rev. Lett. 89 (2002) 200403.

[52] M. Galassi, J. Davies, J. Theiler, B. Gough, G. Jungman, P. Alken, M. Booth, F. Rossi, R. Ulerich, GNU Scientific Library Reference Manual third ed., ISBN 0954612078, 2015, www.gnu.org.

[53] H. Borouchaki, M.J. Castro-Diaz, P.L. George, F. Hecht, B. Mohammadi, 5th Inter. Conf. on Numerical Grid Generation in Computational Field Simulations, Mississipi State Univ., 1996.

[54] M. Castro-Diaz, F. Hecht, B. Mohammadi, Int. J. Comput. Fluid Dyn. 25 (2000) 475–491.

[55] F. Hecht, B. Mohammadi, AIAA Pap. 97 (1997) 0859.

[56] P.L. George, H. Borouchaki, Delaunay Triangulation and Meshing, Hermès, Paris, 1998.

[57] P.J. Frey, Medit: An Interactive Mesh Visualisation Software, RT-0253, INRIA, 2001.

[58] A.W.J. Nocedal, R.A. Waltz, SIAM J. Optim. 19 (4) (2008) 1674–1693.
[59] A. Wächter, L.T. Biegler, Math. Program. 106 (1) (2006) 25–57.
[60] S. Auliac, Développement d'outils d'optimisation pour Freefem++, Thèse, Université Pierre et Marie Curie, Paris, France, 2014.
[61] K.W. Madison, F. Chevy, W. Wohlleben, J. Dalibard, Phys. Rev. Lett. 84 (2000) 806.
[62] V. Bretin, P. Rosenbusch, F. Chevy, G. Shlyapnikov, J. Dalibard, Phys. Rev. Lett. 90 (2003) 100403.
[63] S. Stringari, Phys. Rev. Lett. 82 (1999) 4371.
[64] T. Williams, C. Kelley, Gnuplot 5.0 : An Interactive Plotting Programm, 2015, http://www.Gnuplot.info/.
[65] M. Kilgard, GLUT 3.7, 2000, https://www.opengl.org/resources/libraries/glut/.