FELINE: Finite element solver for hydrodynamic lubrication problems using the inexact Newton method

Alexandre Silva[a], Veniero Lenzi[a], Albano Cavaleiro[b], Sandra Carvalho[b], Luís Marques[a]

[a] *Centre of Physics of Universities of Minho and Porto, University of Minho, Campus de Gualtar, 4710-057, Braga, Portugal*
[b] *University of Coimbra, CEMMPRE - Centre for Mechanical Engineering Materials and Processes, Department of Mechanical Engineering, Rua Luís Reis Santos, 3030-788, Coimbra, Portugal*

Abstract

In this work we present FELINE, a C++ solver of the Reynolds equation for treating hydrodynamic lubrication problems. To correctly describe cavitation regions, FELINE implements the inexact Newton iteration (INE) algorithm within a finite element method (FEM) framework. The solver was tested and validated against known cases in literature and industrially relevant cases of dimpled textures. Furthermore, we provide a benchmark for a complex dimpled texture case to evaluate the performance and robustness of the implementation. FELINE performs very fast when compared with existing implementations and shows a great degree of stability, while providing physically correct solutions thanks to the INE algorithm.

Keywords:
Hydrodynamic lubrication, finite element method, inexact Newton method

Program summary

Program title: FELINE
CPC Library link to program files:
Code Ocean capsule:
Licensing provisions: Creative Commons by 4.0 (CC by 4.0)
Programming language: C++
Nature of problem: A fast and robust solver of the Reynolds equation that solves the linear complementarity problem arising from the treatment of the cavitation boundaries.

*Veniero Lenzi.
E-mail address: veniero.lenzi@fisica.uminho.com

Solution method: An inexact Newton iteration method implemented within a finite element method framework that properly ensures that the complementarity conditions are satisfied by introducing a forcing term.

1. Introduction

The reduction of friction between lubricated parts in contact in mechanical equipment, such as engine pistons and bearings, is fundamental to increase their lifetimes and reduce friction losses, hence reducing their environmental impact [1]. In this sense, research on tribology has developed different approaches such as application of surface coatings [2], research and development of novel lubricants [3] and surface modification techniques [4]. As a matter of fact, the latter approach is widely applied thanks to advances in texturing techniques [5], leading to rapid and efficient pattern generation processes. Despite this, the search of an optimal surface proceeds as a trial-and-error process, and it is still not clear what the best texture might be for a specific application.

Numerical simulations offer a natural way to solve this problem, being significantly cheaper and faster with respect to physical experiments requiring pristine laser textured samples. Because of this, there is a growing effort in numerical simulations [6] to perform the texture optimization of hydrodynamic lubrication problems. These are effectively described by the 2D time-independent Reynolds equation [7], which provides the pressure exerted by the lubricant between two surfaces in relative motion as a function of the surface's profile, relative sliding velocity and applied external load, as well as of the fluid density and viscosity. The main challenge regarding its computational implementation is that cavitation might occur in certain regions: cavitated regions produce no pressure, therefore it is fundamental to represent them correctly by properly describing their boundaries. Moreover, the constraints of zero pressure in cavitated areas and strictly positive pressure in non-cavitated fluid must be imposed to the solutions of the Reynolds equation, so that they are physically sound.

Many strategies have been proposed to solve Reynolds equation with a correct treatment of cavitation. Early attempts ranged from the Jakobsson-Floberg-Olsson (JFO) cavitation model (1957-1965) [8] and the Elrod-Adams cavitation model (1975) [9] to the $\alpha - g$ model [10] which assumed the lubricant film to be a pure liquid phase in non-cavitation regions and a mixture of liquid/gas phases in regions of cavitation. The JFO cavitation model was improved by converting the cavitation boundary into a linear complementarity problem (LCP), named $p - \theta$ model. In [11] and [12] it was shown that the determination of the cavitation region can be cast as a LCP and a method of effective relaxation to update the solution was proposed.

The inexact Newton (INE) method [13] introduces a perturbation parameter meant to prevent the stalling of the algorithm due to the enforcement of the non-negativity condition. The Fischer-Burmeister-Newton-Schur (FBNS) [14] iteration is similar to the INE iteration, however it accepts negative values p and θ, which means that the non-negativity condition is not always strictly guaranteed with FBNS, as opposed to the INE iteration.

From the above discussion it was shown that effective methods have been developed to solve the Reynolds equation with correct treatment of cavitation regions, however

their fast and robust implementation is needed to deal with realistic surfaces and problems of industrial interest. In this work, we present our FELINE solver, which implements the inexact Newton method within a finite element method framework providing a more general and efficient way to treat 2D lubrication problems. The use of FEM allows us to overcome the limitations of the finite differences method, especially in terms of domain shape and mesh construction, hence providing more flexibility. On the other hand, the C++ implementation and use of fast linear algebra libraries (Armadillo [15][16]) allows for an efficient solution of the numerical problem.

2. Methods

2.1. Definition of the problem

In this work the density ρ and viscosity μ of the lubricating fluid are always treated as constants. Moreover, any deformation of the sliding surfaces in contact is neglected as well as any temperature effects. Under these conditions, the problem is defined by the Reynolds equation, which allows us to calculate the lubricant pressure p:

$$\boldsymbol{\nabla} \cdot \left(\rho h^3 \boldsymbol{\nabla} p\right) = 6\mu \, \boldsymbol{U} \cdot \boldsymbol{\nabla} \left(\rho h\right) \ , \tag{1}$$

where h is the film thickness and $\boldsymbol{U}$ the relative sliding velocity, as illustrated in figure 1. Without loss of generality, we can assume that the sliding velocity is always directed along the x axis:

$$\boldsymbol{\nabla} \cdot \left(\rho h^3 \boldsymbol{\nabla} p\right) = 6\mu \, U \frac{\partial(\rho h)}{\partial x} \, . \tag{2}$$

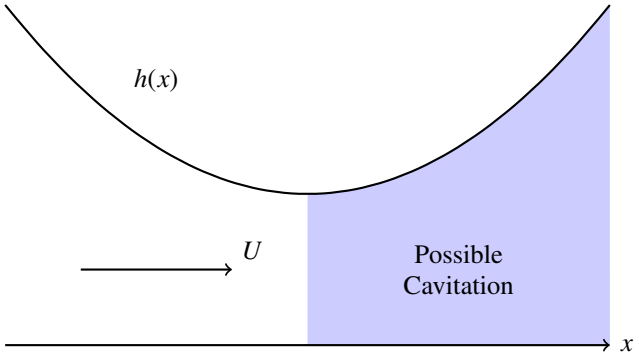


Figure 1: diagram of the film thickness h between sliding surfaces under relative sliding velocity $\boldsymbol{U}$.

In active regions (no cavitation) the pressure is always non-negative and the density ρ is equal to that of the fluid ρ_0. In inactive regions (with cavitation) the pressure is zero and the density is no longer equal to that of the lubricant film ρ_0, since the fluid is a mix of liquid/gas and $p = 0$. In summary:

$$\begin{cases} (\rho_0 - \rho)p = 0 \\ \rho_0 - \rho \geq 0 \\ p \geq 0 \end{cases} \tag{3}$$

Defining the void fraction as $\theta = (\rho_0 - \rho)/\rho_0$ we can rewrite equation (2) as

$$\boldsymbol{\nabla} \cdot \left(h^3 \boldsymbol{\nabla} p\right) + 6\mu\, U \frac{\partial (h\theta)}{\partial x} = 6\mu\, U \frac{\partial h}{\partial x}\,. \tag{4}$$

2.2. Finite elements discretization

To solve equation (4) we seek a discretization of the Reynolds equation and its domain leading to an algebraic system of the form

$$\overline{\overline{\boldsymbol{K}}}_{\boldsymbol{p}} \boldsymbol{p} + \overline{\overline{\boldsymbol{K}}}_{\boldsymbol{\theta}} \boldsymbol{\theta} = \boldsymbol{c}\,, \tag{5}$$

subject to the conditions $\boldsymbol{p} \cdot \boldsymbol{\theta} = \mathbf{0}$, $\boldsymbol{p} \geq \mathbf{0}$ and $\boldsymbol{\theta} \geq \mathbf{0}$.

Equation (5) is a LCP and to solve it we implemented a damped inexact Newton iteration in the finite element method (FEM) framework considering first order triangular elements. The weak formulation of our equation reads

$$\int_\Omega h^3 \boldsymbol{\nabla} w \cdot \boldsymbol{\nabla} p \, d\Omega - 6\mu U \int_\Omega w \frac{\partial (h\theta)}{\partial x}\, d\Omega + 6\mu U \int_\Omega w \frac{\partial h}{\partial x}\, d\Omega - \int_\Gamma w h^3 \, (\boldsymbol{n} \cdot \boldsymbol{\nabla} p)\, d\Gamma = 0\,, \tag{6}$$

where w is the weight or test function, Ω is the domain and Γ is the domain boundary. The boundary integral vanishes for Dirichlet boundary conditions (DBC).

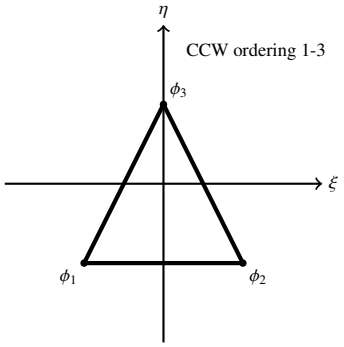
Substituting the approximate solution for p and θ into the weak form of our equation and considering the Galerkin Finite Element Method (GFEM) [17] we obtain a set of equations (5) with coefficients

$$K^p_{ij} = \int_\Omega h^3 \left(\boldsymbol{\nabla} \phi_j \cdot \boldsymbol{\nabla} \phi_i\right) d\Omega \tag{7}$$

$$K^\theta_{ij} = -6\mu U \int_\Omega \left(h \phi_j \frac{\partial \phi_i}{\partial x} + \frac{\partial h}{\partial x} \phi_j \phi_i\right) d\Omega \tag{8}$$

$$c_i = -6\mu U \int_\Omega \phi_j \frac{\partial h}{\partial x}\, d\Omega\,. \tag{9}$$

For a triangular first order element (3 local nodes) with local element coordinates (ξ, η) the basis functions ϕ are defined as follows:



$$\phi_1(\xi, \eta) = \xi$$
$$\phi_2(\xi, \eta) = \eta$$
$$\phi_3(\xi, \eta) = 1 - \xi - \eta$$

Figure 2: Triangular first order element in coordinate space (ξ, η) and respective basis functions.

2.3. Damped inexact Newton iteration

The damped inexact Newton iteration was shown in [13] to be an effective and efficient way to solve the LCP problem. In the following we briefly report the steps of the INE iteration. We can rewrite our set of coupled equations and conditions as

$$\boldsymbol{\Psi}(\boldsymbol{p},\boldsymbol{\theta}) = \mathbf{0} \tag{10}$$

$$\boldsymbol{p} \geq \mathbf{0};\ \boldsymbol{\theta} \geq \mathbf{0}\,, \tag{11}$$

in which

$$\boldsymbol{\Psi}(\boldsymbol{p},\boldsymbol{\theta}) = \begin{pmatrix} \boldsymbol{\Psi}_0(\boldsymbol{p},\boldsymbol{\theta}) \\ \overline{\overline{\boldsymbol{P}}}\,\overline{\overline{\boldsymbol{\Theta}}}\hat{\boldsymbol{e}} \end{pmatrix} = \begin{pmatrix} \overline{\overline{\boldsymbol{K}}}_{\boldsymbol{p}}\boldsymbol{p} + \overline{\overline{\boldsymbol{K}}}_{\boldsymbol{\theta}}\boldsymbol{\theta} - \boldsymbol{c} \\ \overline{\overline{\boldsymbol{P}}}\,\overline{\overline{\boldsymbol{\Theta}}}\hat{\boldsymbol{e}} \end{pmatrix} \tag{12}$$

and where diagonal matrices $\overline{\overline{\boldsymbol{P}}}$ and $\overline{\overline{\boldsymbol{\Theta}}}$ have the values of $\boldsymbol{p}$ and $\boldsymbol{\theta}$ as their entries, and $\hat{\boldsymbol{e}}$ represents the unit vector.

The Newton iteration is defined as

$$\begin{pmatrix} \boldsymbol{p}^{k+1} \\ \boldsymbol{\theta}^{k+1} \end{pmatrix} = \begin{pmatrix} \boldsymbol{p}^{k} \\ \boldsymbol{\theta}^{k} \end{pmatrix} + \alpha^k \begin{pmatrix} \Delta\boldsymbol{p}^{k} \\ \Delta\boldsymbol{\theta}^{k} \end{pmatrix};\ k = 0, 1, 2, ... \tag{13}$$

where $(\Delta\boldsymbol{p}^k, \Delta\boldsymbol{\theta}^k)$ are the updates for each iteration dampened by a factor α_k and are calculated by solving the linear system

$$\boldsymbol{J}^k(\boldsymbol{p},\boldsymbol{\theta}) \begin{pmatrix} \Delta\boldsymbol{p}^{k} \\ \Delta\boldsymbol{\theta}^{k} \end{pmatrix} = -\boldsymbol{\Psi}^k(\boldsymbol{p},\boldsymbol{\theta})\,, \tag{14}$$

As described in [13], a common pitfall with this method is a stalling of the iteration leading to convergence failure. To avoid this, a perturbation term $\eta_k\tilde{\boldsymbol{e}}$ with $\tilde{\boldsymbol{e}} = \begin{pmatrix} \mathbf{0} \\ \mathbf{1} \end{pmatrix}$ is included:

$$\boldsymbol{J}^k(\boldsymbol{p},\boldsymbol{\theta}) \begin{pmatrix} \Delta\boldsymbol{p}^{k} \\ \Delta\boldsymbol{\theta}^{k} \end{pmatrix} = -\boldsymbol{\Psi}^k(\boldsymbol{p},\boldsymbol{\theta}) + \eta_k\tilde{\boldsymbol{e}}\,. \tag{15}$$

The perturbation coefficient η_k is defined as

$$\eta_k = \sigma_k\mu_k\,, \tag{16}$$

where $\sigma_k \in [0, 1]$ is the forcing parameter and μ_k is a parameter in the range

$$\frac{\boldsymbol{p}^k \cdot \boldsymbol{\theta}^k}{N} \leq \mu_k \leq \frac{\left\|\boldsymbol{\Psi}^k(\boldsymbol{p},\boldsymbol{\theta})\right\|}{\sqrt{N}}\,. \tag{17}$$

The choice of relaxation parameter α^k is required to enforce the convergence of the method and this choice is composed of 3 conditions or steps: feasibility, centrality and backtracking.

The feasibility condition provides the initial value of α_k and reads

$$\alpha^k = \min\left(\min_{\Delta p_i^k<0} \frac{-p_i^k}{\Delta p_i^k}, \min_{\Delta\theta_i^k<0} \frac{-\theta_i^k}{\Delta\theta_i^k}, 1\right). \tag{18}$$

This guarantees that conditions $\boldsymbol{p} \geq \mathbf{0}$ and $\boldsymbol{\theta} \geq 0$ are both satisfied after each iteration.

The centrality conditions define the coefficient η_k in the forcing term and ensure $\boldsymbol{p}^k$ and $\boldsymbol{\theta}^k$ are bounded away from $\boldsymbol{p} = 0$ and $\boldsymbol{\theta} = 0$ and are defined as

$$\begin{aligned}
\phi_1^k(\alpha_k) &= \min\left(\boldsymbol{p}^k(\alpha_k) \circ \boldsymbol{\theta}^k(\alpha_k)\right) - \frac{\gamma_k \tau_1}{N} \boldsymbol{p}^k(\alpha_k) \cdot \boldsymbol{\theta}^k(\alpha_k) \geq 0 \\
\phi_2^k(\alpha_k) &= \boldsymbol{p}^k(\alpha_k) \cdot \boldsymbol{\theta}^k(\alpha_k) - \gamma_k \tau_2 \left\| \boldsymbol{\Psi}_0^k(\boldsymbol{p}(\alpha_k), \boldsymbol{\theta}(\alpha_k)) \right\| \geq 0\,,
\end{aligned} \tag{19}$$

where

$$\tau_1^{max} = N \frac{\min\left(\boldsymbol{p}^0 \circ \boldsymbol{\theta}^0\right)}{\boldsymbol{p}^0 \cdot \boldsymbol{\theta}^0}; \quad \tau_2^{max} = \frac{\boldsymbol{p}^0 \cdot \boldsymbol{\theta}^0}{\left\| \boldsymbol{\Psi}_0^0(\boldsymbol{p}, \boldsymbol{\theta}) \right\|} \tag{20}$$

and $\gamma_k \in [0.5, 1.0]$. The centrality conditions also impose a lower bound for the forcing parameter:

$$\sigma_k > \max\left(\delta_k \frac{\sqrt{N} + \tau_1 \gamma_k}{1 - \tau_1 \gamma_k}, \delta_k \frac{\sqrt{N} + \tau_2 \gamma_k}{\sqrt{N}} \right). \tag{21}$$

A recursive reduction of α_k can be used to find a value that satisfies both conditions (19).

The backtracking condition is an additional safeguard that reduces the value of α_k by a factor λ^t with $\lambda \in [0, 1]$ and t a non-negative integer that increases per iteration until the following condition is met

$$\left\| \boldsymbol{\Psi}(\boldsymbol{p}^k + \alpha_k \Delta \boldsymbol{p}^k, \boldsymbol{\theta}^k + \alpha_k \Delta \boldsymbol{\theta}^k) \right\| \leq (1 - \beta \alpha_k (1 - \sigma_k - \delta_k)) \left\| \boldsymbol{\Psi}(\boldsymbol{p}^k, \boldsymbol{\theta}^k) \right\|, \tag{22}$$

where $\beta \in [0, 1]$ and $\sigma_k + \delta_k = 1$. These parameters are set as follows

$$\begin{aligned}
\sigma_k &= \min\left(0.01 \left\| \boldsymbol{\Psi}^k(\boldsymbol{p}, \boldsymbol{\theta}) \right\|, 0.9\right) \\
\delta_k &= \min\left(\frac{\sigma_k}{N \max\left(\frac{\sqrt{N} + \tau_1 + \gamma_k}{1 - \tau_1 \gamma_k}, \frac{\sqrt{N} + \tau_2 \gamma_k}{\sqrt{N}} \right)} \right).
\end{aligned} \tag{23}$$

τ_1	τ_2	μ_k	β	λ	γ_k
$\min\left(0.5\,\tau_2^{max} \times 10^{-7}, 0.99\right)$	$\tau_2^{max} \times 10^{-7}$	μ_k^{max}	10^{-4}	0.5	0.5

Table 1: parameters utilized in the numerical experiments described in [13] for validation of the INE method implementation.

Finally, the stopping criteria for the DIN iteration is defined as

$$\begin{aligned}
c_1 &= \left\| \boldsymbol{\Psi}^{k+1}(\boldsymbol{p}, \boldsymbol{\theta}) \right\| \leq \epsilon_1 \\
c_2 &= \left\| \alpha_k \begin{pmatrix} \Delta \boldsymbol{p}^k \\ \Delta \boldsymbol{\theta}^k \end{pmatrix} \right\| \leq \epsilon_2\,,
\end{aligned} \tag{24}$$

where ϵ_1, ϵ_2 define the convergence threshold of the solver.

The INE solver was implemented in C++17 utilizing the linear algebra library Armadillo [15][16]. The inner solver of choice is SuperLU [18] a sparse direct solver preceded by an equilibration of the sparse Jacobian matrix to help reduce computation times.

3. Results

3.1. Validation

To allow for a direct comparison with the results reported by Mezzadri and Galligani [13], we used their same algorithm parameters, which are reported in table 1. However, we note that those parameters can, and should, be optimized to achieve faster convergence times.

To validate our results, we considered a 2D square domain of length $L = 100\mu m$, a lubricant viscosity $\mu = 0.015\ Pa \cdot s$ and a relative surface sliding velocity $U = 5 \times 10^{-3}\ m \cdot s^{-1}$ and we ran three different test cases matching problems 1-3 of [13]. We can define a Convergent-Divergent (C-D) profile and a Divergent-Convergent (D-C) profile according to the following function

$$h_{\pm}(x, y) = \pm \frac{h_{max} - h_{min}}{2} \sin\left(\frac{2\pi x}{L} + \frac{\pi}{2}\right) + 2\left(h_{max} - h_{min}\right) , \tag{25}$$

with $h_{max} = 0.025\ \mu m$ and $h_{min} = 0.015\ \mu m$, that, when combined with the different boundary conditions, allows us to define three test cases according to table 2.

The following solutions were obtained in a 200×200 element mesh, with 2D and sliced pressure and cavitation profiles for problem 1 shown in figures 3 and 4. Our results are in excellent agreement with profiles of references [13] and [19]. The 2D and sliced results for problem 2 and 3 are shown in figures (5, 6) and (7, 8) respectively, closely matching the 1D profiles of [13]. The difference in pressure magnitudes are due to the fact that, while in [13] these problems are solved in 1D, we are solving them in 2D. We further notice that the cavitation is correctly reproduced where expected, that is in correspondence of divergent profile region.

Problem	Profile	$p_{d\Omega}$	$\theta_{d\Omega}$
1	$h_+(x, y)$	0	0
2	$h_+(x, y)$	1	0
3	$h_-(x, y)$	1	0

Table 2: test case definition where C-D $\equiv h_+(x, y)$, D-C $\equiv h_-(x, y)$ and $p_{d\Omega}$ and $\theta_{d\Omega}$ are the values of p and θ at the domain boundary.

It is also important to highlight that in every solved example the complementarity conditions were obeyed at every iteration, thus providing a robust physical solution. The feasibility condition was sufficient to fulfill the complementarity conditions in these 3 cases, although centrality and backtracking might be necessary to ensure convergence in more complex cases. The cavitation model and inexact Newton iteration method implemented in this work ensure physically realistic solutions.

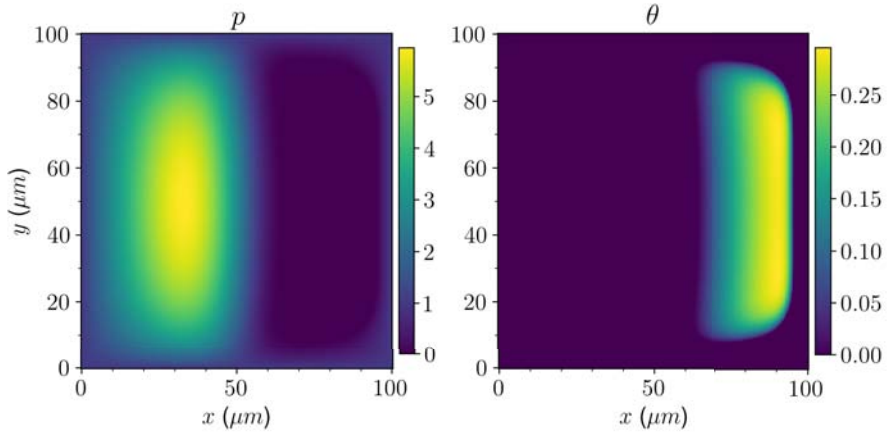


Figure 3: Surface plots of $h/h_{max}(x, y)$, $p(x, y)$ and θ of problem 1 solved in a 200×200 element mesh.

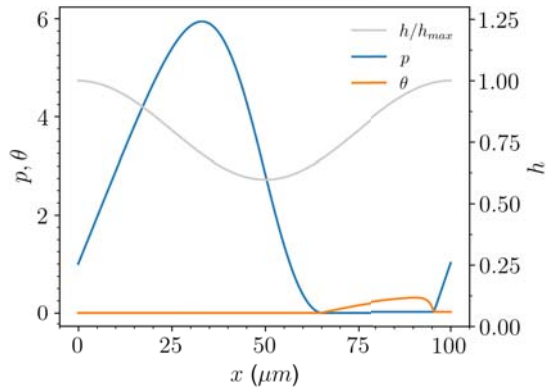


Figure 4: Plots of a slice at $y = 50\ \mu m$ of $h/h_{max}(x, y)$, $p(x, y)$ and θ of problem 1 solved in a 200×200 element mesh.

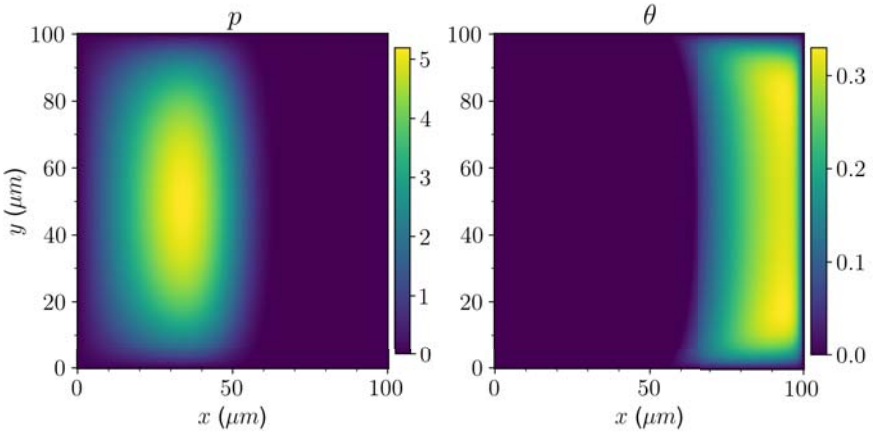


Figure 5: Surface plots of $h/h_{max}(x, y)$, $p(x, y)$ and $\theta(x, y)$ of problem 2 solved in a 200x200 element mesh.

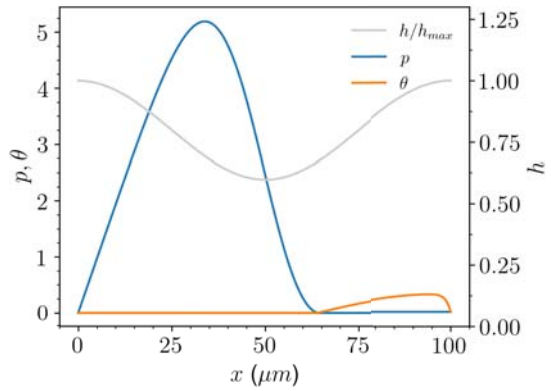


Figure 6: Plots of a slice at $y = 50\ \mu m$ of $h/h_{max}(x, y)$, $p(x, y)$ and $\theta(x, y)$ of problem 2 solved in a 200×200 element mesh.

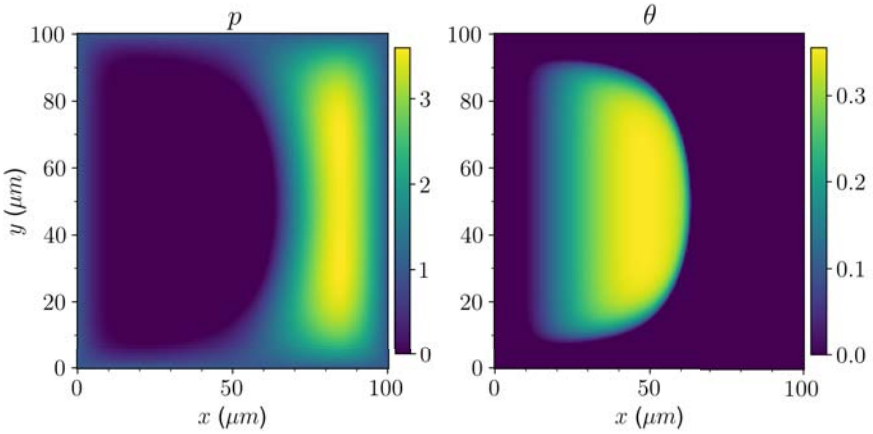


Figure 7: Surface plots of $h/h_{max}(x, y)$, $p(x, y)$ and $\theta(x, y)$ of problem 3 solved in a 200×200 element mesh.

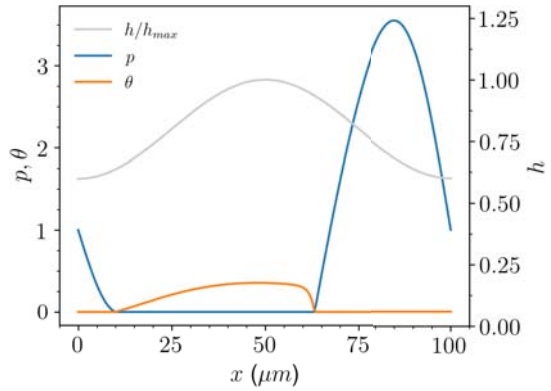


Figure 8: Plots of a slice at $y = 50\ \mu m$ of $h/h_{max}(x, y)$, $p(x, y)$ and θ of problem 3 solved in a 200×200 element mesh.

3.2. Dimpled texture

The code was also tested for a dimpled texture, which is an example of a texture of industrial relevance, For this particular case, we consider a set of 9 equally spaced dimples, with depth $D_d = 10\ \mu m$ and radius $D_r = 60\ \mu m$, placed on a parabolic profile

$$h(x, y) = 0.1 + 2 \times 10^{-5} \cdot (x - 500)^2 \quad (\mu m). \tag{26}$$

Considering a 2D square domain of side $L = 1000\ \mu m$, lubricant viscosity $\mu = 0.035\ Pa \cdot s$ and relative sliding velocity $U = 4.3 \times 10^{-2}\ m \cdot s^{-1}$ the resulting height, pressure, and cavitation 2D profiles and 1D slice at $y = 500\ \mu m$ are shown in figures 9 and 10 respectively.

One can observe that the central cavitation region is diminished along with pressure. Notice also that dimples far from the center have a much lower influence on

pressure as they are relatively far away from the contact region, where the fluid pressure is maximal.

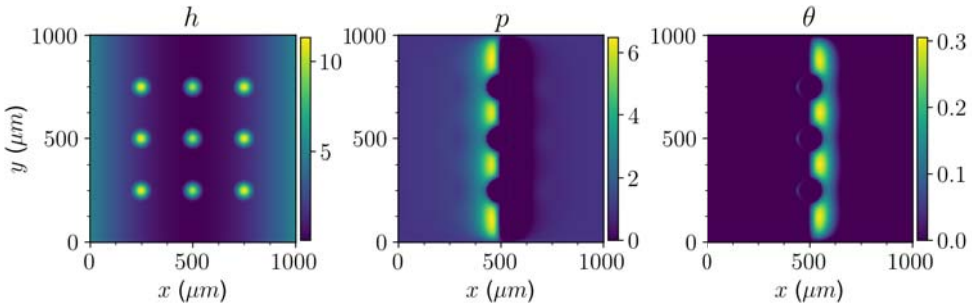


Figure 9: Surface plots of $h/h_{max}(x, y)$, $p/p_{max}(x, y)$ in MPa and θ of the dimpled texture solved in a 200×200 element mesh.

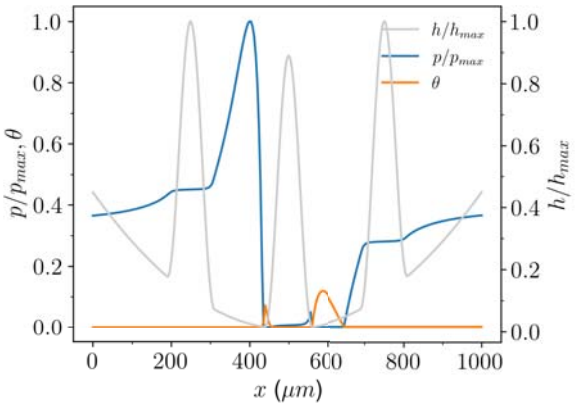


Figure 10: Normalized plots of a slice at $y = 500\ \mu m$ of $h/h_{max}(x, y)$, $p/p_{max}(x, y)$ and θ of the dimpled texture solved in a 200×200 element mesh.

3.3. Inverted dimpled texture

To further verify the robustness of our implementation, we calculated the pressure/cavitation profiles for a set of 9 equally spaced inverted dimples with depth $D_d = 1\ \mu m$ and radius $D_r = 100\ \mu m$ placed on a parabolic profile

$$h(x, y) = 1.05 + 0.2 \times 10^{-5} \cdot (x - 500)^2. \ (\mu m) \tag{27}$$

Considering a 2D square domain of side $L = 1000\,\mu m$, lubricant viscosity $\mu = 0.035\ Pa \cdot s$ and relative sliding velocity $U = 4.3 \times 10^{-1}\ m \cdot s^{-1}$ the resulting normalized height, pressure and cavitation 2D profiles and 1D slice at $y = 500\ \mu m$ are shown in figures 11 and 12 respectively.

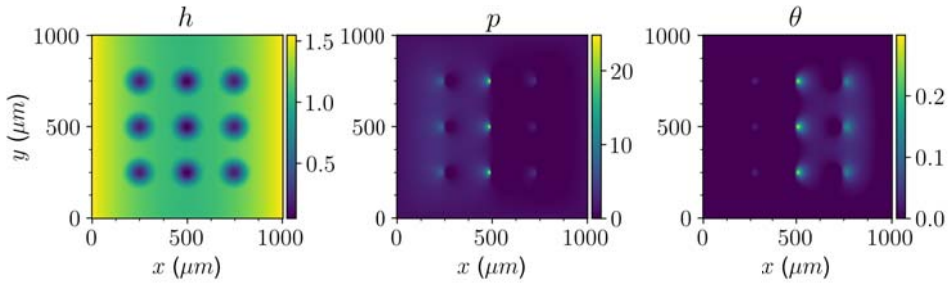


Figure 11: Surface plots of $h/h_{max}(x, y)$, $p/p_{max}(x, y)$ in MPa and θ of the inverted dimpled texture solved in a 200×200 element mesh.

This is typically a hard to solve problem as the inverted dimples introduce very localized contacts between the surfaces as seen in figure 11, hence extremely high pressures at the dimple peaks are found. For each of the inverted dimples, a cavitation region is generated.

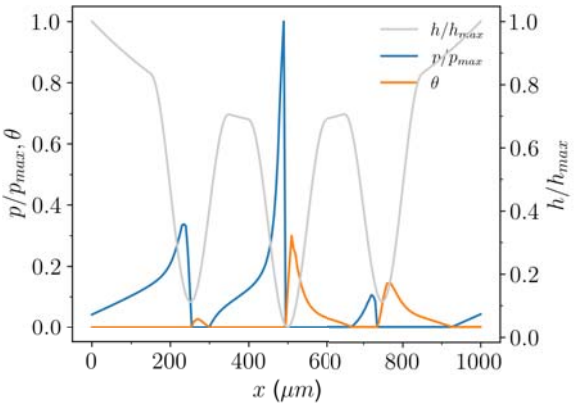


Figure 12: Normalized plots of a slice at $y = 500\ \mu m$ of $h/h_{max}(x, y)$, $p/p_{max}(x, y)$ and θ of the inverted dimpled texture solved in a 200×200 element mesh.

This demonstrates that FELINE is robust enough to successfully solve complex cases, with the centrality condition being called twice and taking around the double of the number of iterations to achieve convergence compared to the previous example in section 3.2.

3.4. Benchmark

To assess the performance of FELINE, we run the code for a dimpled texture case and increasing mesh sizes $N \times N$ on a single core of a Intel(R) Xeon(R) CPU E5-2697 v2. The number of iterations, total execution time and convergence parameters defined in equations (24) are shown in 3.

From the reported times, we can see that the solver is rather fast even if, as expected the computational times increase exponentially with mesh size since the Jacobian matrix scales as N^2. FELINE also shows an impressive 100-fold speedup when compared with the results of a 100×100 mesh reported in [13].

Mesh size	**Total computation time (s)**	**Iterations**	**C_1, C_2 defined in (24)**
50	1.51	19	5.45×10^{-10}, 6.79×10^{-5}
100	6.07	24	4.04×10^{-11}, 2.76×10^{-5}
150	16.8	24	8.36×10^{-12}, 2.76×10^{-5}
200	49.6	26	1.24×10^{-11}, 9.73×10^{-5}
250	87.2	29	4.82×10^{-11}, 6.91×10^{-5}

Table 3: Benchmark of the INE algorithm results utilizing SuperLU for varying levels of mesh coarseness.

The effect of mesh coarseness on the solution correctness was verified. In figure 13 the pressure and cavitation profiles at $y = 500\ \mu m$ is shown for different mesh sizes. The main difference in the pressure profile is in the region at $400 \leq x \leq 600$, where smaller mesh sizes appear to overestimate the pressure within the dimple region, while severely underestimating the first pressure peak at $x = 400$.

Concerning cavitation, the region observed at $x \approx 450$ is completely absent for $N = 50$, while for all the other cases the cavitation region boundaries are matching.

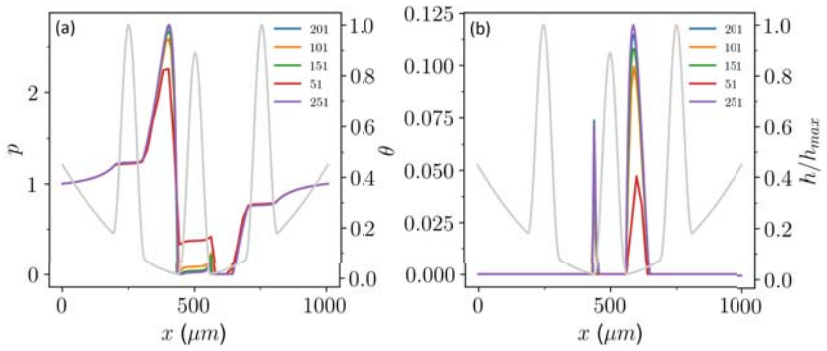


Figure 13: Plots of: (a) a slice at $y = 500\ \mu m$ of $h/h_{max}(x, y)$, $p/p_{max}(x, y)$; (b) a slice at $y = 500\ \mu m$ of $h/h_{max}(x, y)$, $\theta(x, y)$ of the dimpled texture solved in a $N \times N$ element mesh.

4. Conclusions

We developed a fast and efficient finite element method solver for the Reynolds equation with correct treatment of cavitation using the inexact Newton method. It has been validated against known cases in literature and tested with realistic cases such

as dimpled and inverse dimpled textures. In all cases, our solver reproduced almost exactly the results of validation cases and showed a great robustness, capable to provide a converged solution even for the most difficult cases. In terms of speed, we obtained an impressive speedup of around 100 times in comparison to other INE implementations [13].

We clearly demonstrated that our solver can produce fast and accurate results for hydrodynamic lubrication problems, making it a suitable candidate to treat texture optimization problems, where a large number calculations might be needed.

Acknowledgements

This work was supported by the Portuguese Foundation for Science and Technology (in the framework of the Strategic Funding UIDB/04650/2020 project PTDC/EME-SIS/30446/2017 and Advanced Computing Project CPCA/A2/ 4513/2020, awarded by FCT I.P., for accessing MACC-BOB HPC resources.

References

[1] M. Nakada, Trends in engine technology and tribology, Tribology International 27 (1) (1994) 3–8, special Issue Tribology for Automobiles in Japan. doi:https://doi.org/10.1016/0301-679X(94)90056-6.

[2] N. Shaigan, W. Qu, D. G. Ivey, W. Chen, A review of recent progress in coatings, surface modifications and alloy developments for solid oxide fuel cell ferritic stainless steel interconnects, Journal of Power Sources 195 (6) (2010) 1529–1542. doi:https://doi.org/10.1016/j.jpowsour.2009.09.069.

[3] A. E. Somers, P. C. Howlett, D. R. MacFarlane, M. Forsyth, A review of ionic liquid lubricants, Lubricants 1 (1) (2013) 3–21. doi:10.3390/lubricants1010003.

[4] P. Lu, R. Wood, Tribological performance of surface texturing in mechanical applications - a review, Surface Topography: Metrology and Properties 8 (09 2020). doi:10.1088/2051-672X/abb6d0.

[5] I. Etsion, State of the Art in Laser Surface Texturing , Journal of Tribology 127 (1) (2005) 248–253. doi:10.1115/1.1828070.

[6] D. Gropper, L. Wang, T. J. Harvey, Hydrodynamic lubrication of textured surfaces: A review of modeling techniques and key findings, Tribology International 94 (2016) 509–529. doi:https://doi.org/10.1016/j.triboint.2015.10.009.

[7] O. Reynolds, Iv. on the theory of lubrication and its application to mr. beauchamp tower's experiments, including an experimental determination of the viscosity of olive oil, Philosophical transactions of the Royal Society of London (177) (1886) 157–234.

[8] B. Jakobsson, L. Floberg, The finite journal bearing, considering vaporization (das gleitlager von endlicher breite mit verdampfung), 1957.

[9] H. Elrod, A computer program for cavitation and starvation problems, Journal of Lubrication Technology (1975). doi:10.1115/1.3251669.

[10] X.-k. Meng, S.-x. Bai, X.-d. Peng, An efficient adaptive finite element method algorithm with mass conservation for analysis of liquid face seals, Journal of Zhejiang University SCIENCE A 15 (3) (2014) 172–184. doi:10.1631/jzus.A1300328.

[11] R. Ausas, P. Ragot, J. Leiva, M. Jai, G. Bayada, G. C. Buscaglia, The Impact of the Cavitation Model in the Analysis of Microtextured Lubricated Journal Bearings, Journal of Tribology 129 (4) (2007) 868–875. doi:10.1115/1.2768088.

[12] R. F. Ausas, M. Jai, G. C. Buscaglia, A mass-conserving algorithm for dynamical lubrication problems with cavitation, Journal of Tribology 131 (3) (jun 2009). doi:10.1115/1.3142903.

[13] F. Mezzadri, E. Galligani, An inexact newton method for solving complementarity problems in hydrodynamic lubrication, Calcolo 55 (1) (feb 2018). doi:10.1007/s10092-018-0244-9.

[14] T. Woloszynski, P. Podsiadlo, G. W. Stachowiak, Efficient solution to the cavitation problem in hydrodynamic lubrication, Tribology Letters 58 (1) (2015) 18. doi:10.1007/s11249-015-0487-4.

[15] C. Sanderson, R. Curtin., Armadillo: a template-based c++ library for linear algebra., Journal of Open Source Software, Vol. 1, pp. 26 (2016).

[16] C. Sanderson, R. Curtin., A user-friendly hybrid sparse matrix class in c++., Lecture Notes in Computer Science (LNCS), Vol. 10931, pp. 422-430 (2018).

[17] D. Griffiths, J. Lorenz, An analysis of the petrov-galerkin finite element method, Computer Methods in Applied Mechanics and Engineering 14 (1978) 39–64. doi:10.1016/0045-7825(78)90012-9.

[18] X. S. Li, An overview of SuperLU: Algorithms, implementation, and user interface, ACM Trans. Math. Softw. 31 (3) (2005) 302–325.

[19] M. H. Tber, An active-set mixed finite element solver for a transient hydrodynamic lubrication problem in the presence of cavitation, ZAMM Journal of applied mathematics and mechanics: Zeitschrift für angewandte Mathematik und Mechanik 98 (06 2017). doi:10.1002/zamm.201700193.