The purpose of this exercise session is to give an overview of several aspects of scientific computing. First, we focus on the solving of an elliptic boundary value problem with the Finite Element Method. In a first part, we focus on the implementation of a very basic PDE model, hoping this may get the reader to be familiar with the software FreeFem++, which makes it very easy to handle finite element spaces and mesh modifications, within a very few lines of black-box code. Then, we turn to more realistic PDE models from fluid dynamics.

### I. Getting started with FreeFem++

Throughout this section, we will be interested in the numerical resolution of the so-called Poisson problem: \( \Omega \subset \mathbb{R}^d \) being a bounded domain with Lipschitz boundary, \( f \in L^2(\Omega) \) a source term, and \( u_0 \in H^1_0(\partial\Omega) \) a boundary prescription, one searches the solution \( u \) solving:

\[
\begin{aligned}
-\Delta u &= f \quad \text{in } \Omega \\
u &= u_0 \quad \text{on } \partial\Omega
\end{aligned}
\]

(1)

Recall that, from the classical Lax-Milgram theory, this problem admits a unique solution \( u \in H^1_0(\Omega) \), which depends continuously on the data \( f \).

This problem arises, for instance, in electrostatics, as a link between a free charge density \( f \) (up to some constants), and the resulting electric potential field \( u \). Problem (1) enjoys another common physical interpretation, casting \( u \) as the steady state of a heat transfer with source term \( f \).

This first part is devoted to some numerical considerations over problem (1), using the software FreeFem++ [1], a powerful tool for solving boundary value problems with the Finite Element Method. In particular, you will be able to test a whole lot of strategies to solve (1) within very few lines of code.

For numerical issues, use the square \((0,1) \times (0,1)\) as domain \( \Omega \), and \( f = \sin(2\pi x)\sin(2\pi y) \) and \( u_0 = 0 \).

**Question 1:** Define in FreeFem++ the domain \( \Omega \), as well as function \( f \), using listing 1

```
/* Border of the computational domain */
border left(t=0.0,1.0){x = 0.0; y = 1.0-t; label = 1;};
border bottom(t=0.0,1.0){x = t; y = 0.0; label = 1;};
border right(t=0.0,1.0){x = 1.0; y = t; label = 1;};
border top(t=0.0,1.0){x = 1.0-t; y = 1.0; label = 1;};

/* Create, then display initial mesh, with 10 nodes on each part of the boundary */
mesh Th = buildmesh( left(10) + bottom(10) + right(10) + top(10));
plot(Th, wait=1, ps="initial.mesh");

/* Source term for Poisson equation */
func f = sin(2.0*pi*x)*sin(2.0*pi*y);
```

Listing 1. Definition of the data with FreeFem++

**Question 2:** Write the variational formulation associated to problem (1).
• In FreeFem++, introduce a finite element space associated to the mesh defined at question 1. The definition of a finite element space \( V_h \), associated to a mesh \( T_h \), is triggered by the command

\[
\text{fespace } Vh(Th,typeelt);
\]

where \( typeelt \) is the type of the desired finite element; e.g. \( typeelt = P1, P2, \ldots \) for Lagrange-type finite elements, but many other choices are possible.

A typical finite element problem - under variational form - is defined in FreeFem++ through the command

\[
\text{problem nameProblem( } \quad \text{solution and unknown } \quad \in \quad V_h \quad \text{type of solver precision parameter }
\]

\[
\text{int2d(Th)(dx(uh)*dx(vh) ...))} \quad - \quad \text{int2d(Th)}( \ldots )
\]

Left-hand side of variational formula Right-hand side of variational formula

\[
+ \quad \text{on(1,uh=0.0)} \quad ;
\]

boundary conditions on part labelled 1

Such a problem is then solved by simply calling its name as a command:

\[
\text{nameProblem ;}
\]

• What do you observe? Does the problem seem properly solved?

**Question 3**: For increased accuracy in the computation, the most intuitive idea would be to refine the computational mesh \( T_h \). Try solving the same problem, with the same data function \( f \), but with different mesh sizes. Does this work out? Run similar tests with data functions \( f \) showing sharper variations, e.g. functions of the form

\[
f(x,y) = \sin(m\pi x)\sin(n\pi y), \quad m, n \in \mathbb{N}.
\]

**Question 4**: Actually, there is no realistic way to guess *a priori* what would be the ideal size of a mesh \( T_h \) that would enable a proper resolution of a problem such as (1), from the sole knowledge of the data \( f \). Hence the need for *mesh adaptation*.

The basic principle of *mesh adaptation* is to start from a coarse mesh, on which the problem at stake is rather poorly solved (but at a very low CPU cost), and to iteratively infer from increasingly refined meshes and accurate solutions (at still low CPU cost) a ‘good’ mesh, adapted to the solution of the particular problem under consideration (that is, the nodes are concentrated only in the areas where enhanced accuracy is required).

In FreeFem++, there exists a rather sophisticated command

\[
\text{Th = adaptmesh(Th,uh,tol)};
\]

which proceeds to the adaptation of a mesh \( T_h \), with respect to the variations of a function \( u_h \) defined on it, up to a tolerance \( tol \).

Write a loop, which, within say 4 – 5 iterations, proceeds to the adaptation of the coarse initial mesh of question 1 into a mesh amenable to the resolution of problem (1).

II. The heat equation

We now turn to the resolution of the heat equation, which is nothing but the time-dependent version of the previous model. Let \( \Omega \subset \mathbb{R}^2 \) a bounded domain with Lipschitz boundary, filled with a material of thermal conductivity \( \alpha > 0 \), and heated with a source \( f \in L^2(\Omega) \). The temperature distribution at the initial time is described by a function \( u_0 \in H^1(\Omega) \). \( \Omega \) is surrounded by another material with fixed temperature, which is set to 0 for simplicity, so that temperature 0 is imposed at the boundary \( \partial\Omega \) at any time.
Figure 1. (a) Initial mesh (143 points), (b) the associated solution (the amplitude is altogether wrong, although it is not displayed on the figure), (c) adapted mesh, after 5 iterations ($\approx 1000$ points), (d) the associated solution.

The temperature at time $t$ ($0 \leq t \leq T$) in $\Omega$ is then solution to the following equation:

$$
\begin{align*}
\frac{\partial u}{\partial t}(t,x) - \alpha \Delta u(t,x) &= f(x) \quad \text{for} \quad (t,x) \in (0,T) \times \Omega \\
u(t=0,x) &= u_0(x) \quad \text{for} \quad x \in \Omega \\
u(t,x) &= 0 \quad \text{for} \quad t \in (0,T), \ x \in \partial \Omega
\end{align*}
$$

**Question 1:** Write down the variational formulation for (2), at the continuous level in space and time (i.e. the time derivative in (2) is not yet discretized). This is achieved by:

1. identifying the functional space $V$ to which each function $u(t,.)$ should belong to,
2. multiplying the first equality in (2) by a (time-independent) function $v \in V$,
3. performing integration by parts where it is needed to end with formulae as close as possible to those of section I.

**Question 2:** The sequence $t \mapsto u(t,.)$ is approximated by a series of functions $u^n(x) \approx u(t^n,x) \in V$, where $0 < t^0 < \ldots < t^N = T$ is a subdivision of the time interval $(0,T)$, with time step $\Delta t$: $t^n = n\Delta t$, $n = 0, \ldots, N$. Choose a finite difference discretization for $\frac{\partial u}{\partial t}(t^n,.)$ in terms of $u^{n-1}, u^n, u^{n+1}$. Note that several choices may be possible.

**Question 3:** Derive the discrete-in-time, space-continuous variational formulation for each function $u^n$ (involving $u^{n-1}, \ldots$).
**Question 4:** Time to go to numerics! In this example (see 2), the domain \( \Omega \) is the rectangle \((0, 2) \times (0, 1)\), and the final time is \( T = 5 \). The thermal conductivity of the considered material is \( \alpha = 0.1 \), the source \( f \) is concentrated in a small ball around point \( p_1 = (0.5, 0.75) \) of the domain:

\[
f(x) = \begin{cases} 
1 & \text{if } ||x - p_1|| < 0.12 \\
0 & \text{otherwise}
\end{cases}
\]

and the initial distribution of temperature \( u_0 \) inside \( \Omega \) is given by:

\[
u(x) = \begin{cases} 
100(0.2 - ||x - p_2||) & \text{if } ||x - p_2|| < 0.2 \\
0 & \text{otherwise}
\end{cases}
\]

where \( p_2 = (1.5, 0.25) \).

![Figure 2. Initial and boundary conditions for the proposed test case for the numerical resolution of the heat equation (2).](image)

Implement the variational formulation of question 3. into **FreeFem++**, and run several tests:
- using different kinds of finite elements for the approximation of functions \( u^n, n = 0, ..., N \) (e.g. \( P^1 \), \( P^2 \) Lagrange finite elements).
- using various finite difference approximations of the time-derivative \( \frac{\partial u}{\partial t} \) (see question 2).
- using different values for the time \( \Delta t \).
- acting on the mesh of the domain (for instance, try to use the mesh adaptation technique presented in Section I at some iterations of the process).

What do you observe? What is the influence of these different ways of tackling the problem?

**III. Potential flow around an airfoil**

The purpose of this section is to study the flow of the atmosphere around a NACA airfoil, in the simplifying approximation of potential flows. Recall that, in 2d, the velocity field \( u \) of an incompressible fluid, such as the atmosphere at low Mach number, is described by a stream function \( \psi \), such that:

\[
u = \text{curl}(\psi) := \left( \frac{\partial \psi}{\partial y}, -\frac{\partial \psi}{\partial x} \right).
\]

The velocity field \( u \) is then tangent to the isolines of this stream function. Using the irrotationality assumption yields the following equation over \( \psi \):

\[-\Delta \psi = 0.
\]

We now consider a NACA airfoil \( D \), which has a symmetric structure with respect to the horizontal axis, and whose half-thickness \( h(x) \) is given by the equation:

\[
h(x) = 0.17735\sqrt{x} - 0.075597x - 0.212836x^2 + 0.17363x^3 - 0.06254x^4.
\]
\(D\) is embedded in the ball \(\Omega\) of center 0 and radius 5, approximating the full two-dimensional space (see Figure 3).

![Figure 3. A NACA airfoil embedded in a large ball approximating the full plane.](image)

The system under scrutiny is then:

\[
\begin{align*}
-\Delta \psi(x) &= 0 \quad \text{for} \quad x \in \Omega \setminus D \\
\psi(x) &= y \quad \text{for} \quad x \in \partial \Omega \\
\psi(x) &= 0 \quad \text{for} \quad x \in \partial D,
\end{align*}
\]

where the boundary condition imposed on \(\partial \Omega\) mimicks an horizontal velocity profile.

**Question 1:** Write down the variational formulation associated to equation (3).

**Question 2:** Implement the variational formulation of question 1 in FreeFem++, adapting the computational mesh around the boundary of the airfoil, and with respect to the values of the stream function \(\psi\).

**IV. The stationary Stokes equations**

In this section, we are interested in the stationary Stokes equations, posed in a domain \(\Omega \subset \mathbb{R}^2\) with Lipschitz boundary: a fluid with dynamic viscosity \(\nu\) occupying \(\Omega\) is described by its velocity \(u : \Omega \to \mathbb{R}^2\), and pressure field \(p : \Omega \to \mathbb{R}^2\), which satisfy the system:

\[
\begin{align*}
-\nu \Delta u(x) + \nabla p(x) &= f(x) \quad \text{for} \quad x \in \Omega \\
\text{div} \, u(x) &= 0 \quad \text{for} \quad x \in \Omega \\
u(x) &= u_d \quad \text{for} \quad x \in \partial \Omega,
\end{align*}
\]

where \(u_d\) is a prescribed velocity field on \(\partial \Omega\), and \(f\) is a source term.

**Question 1:** Write down the variational formulation associated to equation (4).

Consider, in FreeFem++, the classical *lid driven cavity* for Stokes problem: \(\Omega\) is the unit square \((0,1) \times (0,1),\)
filled with a fluid of unit dynamic viscosity: \( \nu = 1 \). The imposed velocity field \( u_d \) on \( \partial \Omega \) is:

\[
\forall x \in \partial \Omega, \quad u_d(x) = \begin{cases} 
(1, 0) & \text{if } x \text{ lies on the upper part of } \partial \Omega \\
(0, 0) & \text{otherwise}
\end{cases}
\]

and the source term is set to \( f = 0 \).

We are interested in a particular form for the meshes of \( \Omega \), that of a Cartesian grid split into triangles. In FreeFem++, such a mesh \( \mathbf{T} \) is generated using the following command:

\[
\mathbf{T} \mathbf{h} = \text{square}(50, 50);
\]

Using this command, the labels of the boundary \( \partial \Omega \) are automatically set as depicted in figure 4.

![Figure 4. A particular mesh of the unit square, and the boundary labels generated by the FreeFem++ command square.](image)

**Question 2:** Implement the variational formulation of question 1. in FreeFem++, using \( P^2 \) Lagrange finite elements for discretizing the velocity \( u \), and \( P^1 \) elements for the pressure.

*Hint: use the penalization method (seen in the theoretical part of the course) to bring back the variational formulation of question 1. to a ‘classical’ variational formulation over the couple \( (u, p) \).*

**Question 3:** Try out other pairs of finite elements for discretizing \( (u, p) \), and notably the pairs \( P^1 \times P^0 \) and Bubble-\( P^1 \times P^0 \). Does the method always produce the ‘correct’ solution to the system? Why is that so, according to you?

**V. The unsteady Stokes equations**

We are now getting closer and closer to realistic models; let us consider the transient model associated to Stokes equations (4). A fluid, whose density is set to \( \rho = 1 \) for simplicity, with dynamic viscosity \( \nu \), is occupying a domain \( \Omega \), and is described by means of its (time-dependent) velocity field \( u(t, x) \) and pressure field \( p(t, x) \) all along a period of time \( (0, T) \).

The initial state of the fluid is known, that is, \( u(t = 0, .) = u_0, \) and \( p(t = 0, .) = p_0 \) are given. Mixed boundary conditions are considered, that is, \( \partial \Omega \) is decomposed into two complementary parts \( \Gamma_D \) and \( \Gamma_N \). The velocity \( u(t, .) = u_d \) is prescribed at every time on \( \Gamma_D \), and an external stress \( g \) is exerted on \( \Gamma_N \), so
that the system is driven by the system of equations:

\[
\begin{align*}
\frac{\partial u}{\partial t}(t,x) - \nu \Delta u(t,x) + \nabla p(t,x) &= f(t,x) \quad \text{for} \quad (t,x) \in (0,T) \times \Omega \\
\text{div } u(t,x) &= 0 \quad \text{for} \quad (t,x) \in (0,T) \times \Omega \\
u \frac{\partial u}{\partial n}(t,x) - p(t,x) \cdot n(x) &= g(t,x) \quad \text{for} \quad t \in (0,T) \text{ and } x \in \Gamma_D \\
u u(0,x) &= u_0(x) \quad \text{for} \quad x \in \Omega.
\end{align*}
\]

**Question 1:** Write down the variational formulation associated to system (5).

We now consider the benchmark obstacle test case, described in figure 5: on the left (blue) part of the boundary, a parabolic profile is imposed for the velocity: \( u_d(x) = 4y(1-y) \), whereas on the remainder of the Dirichlet boundary (in black), homogeneous Dirichlet boundary conditions are imposed: \( u_d(x) = 0 \). Eventually, the right, red part corresponds to the Neumann part \( \Gamma_N \) of the boundary, and no stress is applied: \( g = 0 \). Similarly, no body forces are applied: \( f = 0 \).

The other parameters for the computation are: \( \nu = 0.005 \), \( T = 40 \) and

\[
\forall x \in \Omega, \quad u_0(x) = \begin{cases}
(4y(1-y),0) & \text{if } x \text{ lies on blue part of } \partial \Omega \\
(0,0) & \text{otherwise}
\end{cases}.
\]

**Figure 5.** The obstacle test case for the unsteady Stokes system.

**Question 2:** Implement the previous variational formulation in **FreeFem++**. Once again, try different combinations of finite element spaces for the velocity and pressure fields, as well as different discretizations of the time derivative \( \frac{\partial u}{\partial t} \). You may also try using the mesh adaptation function described in section I. to reach a higher quality result.

**VI. Darcy’s equation for porous media**

In this section, we investigate the motion of a fluid flowing through the porous domain \( \Omega \) depicted on Figure 6; in other words, \( \Omega \) is composed of a medium whose microscopic structure consists of infinitesimally small grains of solid matter.

As usual, the fluid is characterized by its (vector) velocity \( u \), and its (scalar) pressure field \( p \), and the boundary of \( \Omega \) decomposes into three disjoint parts: \( \partial \Omega = \Gamma_{in} \cup \Gamma_{out} \cup \Gamma_{sl} \). In this context, the couple \( (u,p) \) is the solution to Darcy’s equations:

\[
\begin{align*}
\begin{cases}
    u(x) = K(f(x) - \nabla p(x)) & \text{in } \Omega, \\
    \text{div } u(x) = 0 & \text{in } \Omega, \\
    u(x) = u_{in} & \text{on } \Gamma_{in}, \\
    u(x) \cdot n(x) = 0 & \text{on } \Gamma_{sl}, \\
    p(x) = p_{out} & \text{on } \Gamma_{out}.
\end{cases}
\end{align*}
\]

where the velocity \( u_{in} \) entering \( \Omega \) is \( u_{in}(x) = 4x(1-x) \), the output pressure is \( p_{out} = 1.0 \), and the gravity \( f \) is set to 0 for simplicity. \( K \) is a \( 2 \times 2 \) matrix called the permeability tensor of the medium; it encodes the microscopic structure of the porous medium.
**Figure 6.** Setting for the study of a fluid flowing through a porous medium, whose microscopic composition of the medium is exemplified.

**Question 1:** Write down the variational formulation associated to Darcy’s system (6). Observe that, contrary to the case of Stokes’s equation, the framework of Darcy’s system allows to fix the pressure on some parts of the boundary $\partial \Omega$.

**Question 2:** Implement the variational formulation of the previous question in FreeFem++, trying out several values for the permeability tensor $K$ (in particular, you may be interested in simulating the case of an isotropic medium, e.g. $K = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, and the case of an anisotropic medium, e.g. $K = \begin{pmatrix} 0.1 & 0 \\ 0 & 1 \end{pmatrix}$). Is it possible to use any pair of finite element spaces for the couple $(u, p)$?

### VII. Brinkman’s penalization law

The purpose of this section is to present an alternative system of equations for the motion of a fluid through a porous medium, which also provides a very convenient trick when it comes to performing numerical simulations on domains which are not necessarily discretized by a mesh.

Consider the situation described on Figure 7: a viscous fluid, with viscosity $\nu = 1.0$ flows through a channel $\Omega$, a subdomain $\Omega_p$ of which contains a porous medium (the greyed region on Figure 7).

The velocity $u$ of the fluid and its pressure $p$ satisfy the Brinkman law:

\[
\begin{cases}
-\nu \Delta u(x) + \nabla p(x) + \mu M(x)u(x) = 0 & \text{in } \Omega, \\
\text{div } u(x) = 0 & \text{in } \Omega, \\
u(x) = u_{in} & \text{on } \Gamma_{in}, \\
p(x) = p_{out} & \text{on } \Gamma_{out}, \\
u(x) = 0 & \text{on } \partial \Omega \setminus (\Gamma_{in} \cup \Gamma_{out}),
\end{cases}
\]

where $u_{in}(y) = 4y(1-y)$, and the output pressure $p_{out}$ is set to 1. In this equation, the tensor $M$ plays the role of the inverse of a permeability tensor. For simplicity, we assume it is isotropic, and:

$M(x) = 0$ if $x \in \Omega \setminus \overline{\Omega_p}$, and $M(x) = m$ constant in $\Omega_p$.

Roughly speaking, the larger the value of $m$, the more impervious the region $\Omega_p$ to the motion of the fluid.

**Question 1:** Establish the variational formulation associated to the Brinkman law (7).
Implement the variational formulation of the previous question in FreeFem++, and try out several values for the inverse permeability $m$ in the region $\Omega_p$ (you may be especially interested in trying large values).

**Question 3:** Simulate Stokes’ equations in the domain $\Omega \setminus \overline{\Omega_p}$, imposing no-slip boundary conditions on the boundary $\partial \Omega_p$. What do you observe?

**VIII. A stationary bifluid problem**

In this section, we are interested in a bifluid problem, making the simplifying assumption that an equilibrium state has been achieved, which implies in particular that the interface between both fluids is fixed.

More precisely, let $D$ be a working domain; an ‘interior’ subdomain $\Omega^0 \subseteq D$ of $D$ contains a fluid with dynamic viscosity $\nu^0$, and the complementary, ‘exterior’ set $\Omega^1 = D \setminus \overline{\Omega^0}$ is filled by a different fluid with viscosity $\nu^1$. Let $\Gamma = \partial \Omega^0$ denote the interface between the two phases. The system is subject to Stokes bifluid system:

$$
\begin{align*}
-\nu^i \Delta u^i(x) + \nabla p^i(x) &= f^i(x) & \text{for } x \in \Omega^i, i = 0, 1, \\
\text{div } u^i(x) &= 0 & \text{for } x \in \Omega^i, i = 0, 1, \\
u^0(x) - u^1(x) &= 0 & \text{for } x \in \Gamma, \\
(\sigma^0(x) - \sigma^1(x)) \cdot n^0(x) &= -\gamma \kappa(x)n^0(x) & \text{for } x \in \Gamma, \\
u^1(x) &= u_d & \text{for } x \in \partial D
\end{align*}
$$

where $u^i, p^i$ and $f^i$ denote respectively the velocity, pressure, and body forces applied in the part of the fluid lying in $\Omega^i$, $i = 0, 1$, $\gamma > 0$ stands for the surface tension coefficient, $\kappa(x)$ is the mean curvature of $\Gamma$, and $n^0$ is the unit normal vector to $\Gamma$, pointing outward $\Omega^0$.

**Question 1:** Write down the variational formulation associated to equation (4).

As a numerical example, we consider *Couette’s problem* (see Figure 8): the working domain $D$ is the unit square, and the interior subdomain $\Omega^0$ is the centered disk with radius 0.1 (hence, the curvature of the interface $\Gamma$ is constant, equal to $\kappa = 10$). The physical parameters of the fluids are $\gamma = 0.1$, $\nu^0 = 1$, $\nu^1 = 0.001$, and the imposed velocity $u_d$ on $\partial D$ is:

$$
\forall x \in \partial D, \quad u_d(x) = \begin{cases} 
(1, 0) & \text{if } x \text{ lies on the upper part of } \partial D, \\
(-1, 0) & \text{if } x \text{ lies on the lower part of } \partial D, \\
(0, 0) & \text{otherwise.}
\end{cases}
$$

**Question 2:** Implement the previous variational formulation in FreeFem++. You may be interested in trying several different sets of values as for $\gamma, \nu^0, \nu^1$. 

---

**Figure 7.** Setting for the study of a fluid flowing governed by the Brinkman law.
IX. The Navier-Stokes equation by the method of characteristics

We now come to the central model in this course, namely that of Navier-Stokes equations, which is a more realistic description of the motion of an incompressible fluid than that of the mere Stokes equations. Using the exact same notations as in the previous section, they characterize the couple velocity-pressure \((u, p)\) of an incompressible fluid as the solution to:

\[
\begin{align*}
\frac{∂u}{∂t}(t, x) + u \cdot \nabla u(t, x) - ν \Delta u(t, x) + \nabla p(t, x) &= f(t, x) \quad \text{for } (t, x) ∈ (0, T) × Ω \\
\text{div } u(t, x) &= 0 \quad \text{for } (t, x) ∈ (0, T) × Ω \\
(u(t, x)) &= u_d(x) \quad \text{for } t ∈ (0, T) \text{ and } x ∈ Γ_D \\
ν \frac{∂u}{∂n}(t, x) - p(t, x)n(x) &= g(t, x) \quad \text{for } t ∈ (0, T) \text{ and } x ∈ Γ_N \\
u_0(x) &= u_0(x) \quad \text{for } x ∈ Ω.
\end{align*}
\]

The main difference between Stokes and Navier-Stokes equations lies in the nonlinear convective term \(u \cdot \nabla u\), induced by the acceleration of the fluid. This term can be neglected when the velocity of the fluid is low when compared to the viscous forces, but may become dominant in the high-velocity regime, in which the Stokes approximation becomes dramatically rough. This term is also the main cause of the difficulties in the theoretical and numerical treatments of Navier-Stokes equations.

In this section, we propose to solve equation (9) using the method of characteristics for dealing with the (difficult) convective term.

If \(x_0 ∈ Ω\), and \(t_0 ∈ T\), denote as \(t ↦ X(t, t_0, x_0)\) the characteristic curve passing at \(x_0\) at time \(t = t_0\), solution to the ordinary differential equation:

\[
\begin{align*}
∀ \ t ∈ (0, T), \quad \dot{X}(t, t_0, x_0) &= u(t, X(t, t_0, x_0)) \\
X(t_0, t_0, x_0) &= x_0
\end{align*}
\]

which is nothing but the trajectory of a fluid particle located at \(x_0\) at time \(t_0\). Then, one can see that:

\[
\frac{d}{dt}(u(t, X(t, t_0, x_0))) = \left(\frac{∂u}{∂t} + u \cdot \nabla u\right)(t, X(t, t_0, x_0)),
\]

meaning that the convective term is the time derivative of \(u\) along the trajectories of the fluid particles, in other terms the dynamic acceleration of the fluid. According to this remark, the following finite difference
approximation can be proposed for the convective term (with obvious notations):
\[
\left( \frac{\partial u}{\partial t} + u \cdot \nabla u \right)(t^n, x) \approx \frac{u^n(x) - u^{n-1}(X(t^{n-1}, t^n, x))}{\Delta t}
\]

Now, in the course of an iterative computation, if \( u^{n-1} \) is available, one only has to evaluate the term \( u^{n-1}(X(t^{n-1}, t^n, x)) \). This can be achieved in FreeFem++ using the \texttt{convect} command (see listing 2).

\begin{verbatim}
Vh ux, uy; // components of the velocity field
real dt = 0.05; // time step
Vh f, g; // f= scalar quantity to be convected, g = resulting quantity
f = convect([ux, uy], -dt, f); // sign - indicates backtracking of the
// characteristic curves
\end{verbatim}

Listing 2. Backtracking a characteristic curve in FreeFem++

Consider the \textit{step} test case, depicted in figure 9.

![Figure 9. The step test case for the Navier-Stokes equations.](image)

Here, \( T = 40, \nu = 0.005 \), and the boundary conditions are exactly those of the previous example, and the initial velocity field is given by:

\[ \forall x \in \Omega, \quad u_0(x) = \begin{cases} 
(4y(1-y),0) & \text{if } x < 2 \\
(0,0) & \text{otherwise}
\end{cases} \]

Question: Implement the Navier-Stokes model in FreeFem++, relying on the method of characteristics for the discretization of the convective term.

X. The Rayleigh-Taylor instability

The purpose of this section is to study the so-called \textit{Rayleigh-Taylor instability}, an instability occurring at the interface separating two fluids with different densities.

The situation is that depicted in Figure 10: a box \( D \) is filled with two fluids, occupying the respective subdomains \( \Omega^0, \Omega^1 \subset D \); the interface between both fluids is denoted by \( \Gamma \), and the physical properties of each phase (velocity, pressure, density, etc...) are respectively denoted with a \textsuperscript{0} or \textsuperscript{1} exponent.

The initial position \( \Gamma(0) \) of the interface is depicted on Figure 10: it is provided by the equation

\[ y(x) = 1.5 + 0.1 \cos \left( \frac{\pi}{2} + \pi x \right) , \]

and then evolves in time according to the velocity of the fluid (which satisfies the continuity condition \( u^0(t, x) = u^1(t, x) \) for \( x \in \Gamma(t) \)).
The velocity and pressure \((u^i, p^i)\) of the two fluids are governed by the Navier-Stokes equations:

\[
\begin{align*}
\rho^i(t, x) \left( \frac{\partial u^i}{\partial t}(t, x) + u^i(t, x) \cdot \nabla u^i(t, x) \right) - \nu^i \Delta u^i(t, x) + \nabla p^i(t, x) &= \rho^i(t, x) f(x) \quad \text{for} \quad x \in \Omega^i, \\
\text{div} u^i(t, x) &= 0 \quad \text{for} \quad x \in \Omega^i, \\
u^0(t, x) - u^1(t, x) &= 0 \quad \text{for} \quad x \in \Gamma(t), \\
(\sigma^0(t, x) - \sigma^1(t, x)) \cdot n^0(t, x) &= 0 \quad \text{for} \quad x \in \Gamma(t), \\
u(t, x) n(x) &= 0 \quad \text{for} \quad x \in \Gamma_{sl}, \\
u(t, x) &= 0 \quad \text{for} \quad x \in \Gamma_D.
\end{align*}
\]

Note that, for the sake of simplicity, contrary to the case at stake in Section VIII, the surface tension phenomenon at the interface \(\Gamma(t)\) has been ignored.

**Question 1:** Write down the variational formulation for system (10), getting inspired from what you did in Section VIII.

In the numerical example of interest, the viscosities \(\nu^0, \nu^1\) of both fluids have the shared value 0.0001. The fluid filling \(\Omega^0\) is three times denser than that filling \(\Omega^1\): \(\rho^0 = 3, \rho^1 = 1\), and the only force at play is the gravity \(f = (0, -0.1)\). As for the time step \(dt\), you may take \(dt = 0.5\).

**Question 2:** Use the method of characteristics, as you did in Section IX to simulate the motion of the interface \(\Gamma(t)\). In doing so, at each iteration of the process, you will need to move each node of the mesh according to the velocity of the fluid. To achieve this, you may take advantage of the FreeFem++ movemesh and checkmovemesh commands, whose uses are exemplified in Listing 3 below (see the FreeFem++ documentation for more details).
real dt = 0.05; // time step
real chkm; // Quality of the worst element in the mesh
real EPS = 1.e-10; // Minimum authorized quality for an element in the mesh

chkm = checkmovemesh(Th, [x+dt*ux, y+dt*uy]); // worst quality of an element in the mesh whose elements have been moved according to x \rightarrow x + u_x, y \rightarrow y + u_y

if ( chkm > EPS ) {
    Th = movemesh(Th, [x+dt*ux, y+dt*uy]); // Perform the motion of the mesh when its validity has been checked
    Th = adaptmesh(Th, hmin = 0.01, hmax=0.01); // Remeshing to improve the quality of the mesh
} else {
    cout << "Problem" << endl;
}

Listing 3. Moving a mesh in FreeFem++

Remark This problem is notoriously difficult to simulate; one reason is that the interface $\Gamma(t)$ tends to develop singularities as time passes, which makes the mesh movement operation difficult (you may experience mesh tangling difficulties in trying out different sets of parameters).

XI. The Navier-Stokes equation using Newton-Raphson’s algorithm

In this section, we present an application of a general method for solving nonlinear PDE to the stationary Navier-Stokes equation, namely Newton-Raphson algorithm. Its contents can be adapted to the case of the time-dependent Navier-Stokes equation, or to various nonlinear PDE such as the nonlinear elasticity system, etc...

The general, abstract framework for this method proceeds as follows: let $V$ be a vector space (e.g. a Hilbert space), and consider the problem:

$$\text{(11)} \quad \text{Find } u \in V \text{ s.t. } F(u) = 0,$$

for a given (smooth) function $F : V \rightarrow V$.

The method starts from an initial guess $u_0$ as for the desired root $u$ of $F$ which is ‘not so far’ from $u$. Then, (11) is linearized around $u_0$, which leads to the following approximate problem:

$$\text{Find } u_1 \in V \text{ s.t. } F(u_0) + F'(u_0)(u_1 - u_0) = 0,$$

which produces, under adequate hypotheses on the invertibility of $F'(u_0)$:

$$u_1 = u_0 - F'(u_0)^{-1}F(u_0).$$

Now, $u_1$ is not yet the desired root $u$ of $F$, but is hopefully closer from it than $u_0$ was. The process can be iterated by linearizing (11) anew, around $u_1$, giving rise to a new element

$$u_2 = u_1 - F'(u_1)^{-1}F(u_1),$$

and eventually to a sequence $\{u_n\}$ converging to the desired $u \in V$.

Now, let us apply this general idea to the case of the stationary Navier-Stokes equation:

$$ \begin{cases}
    u.\nabla u(x) - \nu \Delta u(x) + \nabla p(x) = f(x) & \text{for } x \in \Omega \\
    \text{div } u(x) = 0 & \text{for } x \in \Omega \\
    u(x) = u_d(x) & \text{for } x \in \Gamma_D \\
    \nu \frac{\partial u}{\partial n}(x) - p(x)n(x) = g(x) & \text{for } x \in \Gamma_N
  \end{cases} \tag{12}$$

Question 1: Write down the variational formulation for system (12) under the form:

Find $(u, p) \in V \times W$ s.t. $\forall (v, q) \in V_0$, $W_0$,

$$a(u, u, v) + b_1(u, v) + b_2(u, q) + c_1(v, p) + c_2(p, q) = \ell_1(v) + \ell_2(q),$$

13
where $V$ is a functional space to be found, $V_0$ is the associated homogeneous space, $b_1, b_2, c_1, c_2$ are bilinear, $\ell_1, \ell_2$ are linear, and $a$ is trilinear, i.e. $u \mapsto a(u, u, v)$ is nonlinear (it is the term that encloses all the nonlinearity of (12)).

**Question 2:** Adapt Newton-Raphson’s idea to this situation to end up with a scheme of the form:

- Start from $(u_0, p_0) \in V$,
- For $n = 1, \ldots$, find $(\delta u_n, \delta p_n) \in V$ such that:
  \[
  \forall (v, q) \in V_0, \quad a(\delta u_n, u_{n-1}, v) + a(u_{n-1}, \delta u_n, v) + b_1(\delta u_n, v) + b_2(\delta u_n, q) \\
  + c_1(v, \delta p_n) + c_2(\delta p_n, q) = \ell_1(v) + \ell_2(q).
  \]
- Get $u_n, p_n$ as: $u_n = u_{n-1} + \delta u_n$, and $p_n = p_{n-1} + \delta p_n$.

**Question 3:** Implement this numerical process in FreeFem++, by using, e.g., the test-case of the previous section.

**Final Remark** Numerous different techniques exist for the solution of the Navier-Stokes equations, among which Chorin-Temam’s projection method. At this point, you should have all the required knowledge about FreeFem++ for programming them!

**References**
