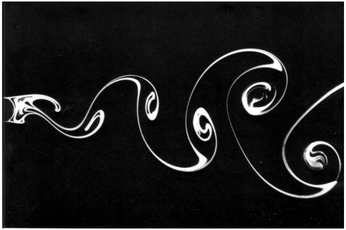
Specialized Numerical Methods for Transport Phenomena

The finite element method: Navier-Stokes equations



Bruno Blais

Associate Professor
Department of Chemical Engineering
Polytechnique Montréal

February 26, 2024

Outline

Outline

Motivation

We have finally reached the pinnacle of where we want to be concerning classical methods: the solution of the incompressible Navier-Stokes equations.

These equations are key to engineering. They describe multiple phenomena:

- Flow of gases (as long as $\mathrm{Ma} < 0.3$). Wind turbines, gas in chemical processes, pneumatic transport, etc.
- Flow of liquids. Blood, microfluidics, heat exchangers, molten plastic, etc.
- It's hard to find an industry in which they are not important.

Let's make sure we fully understand them before we try to solve them...

Navier-Stokes equations

Problem definition

$$\nabla \cdot \boldsymbol{u} = 0$$

$$\rho \frac{\partial \boldsymbol{u}}{\partial t} + \rho \left(\boldsymbol{u} \cdot \nabla \right) \boldsymbol{u} = -\nabla p + \mu \nabla^2 \boldsymbol{u} + \rho \boldsymbol{g}$$

- $\mathbf{u}$ is the velocity vector.
- p is the pressure.
- $\rho = C$ is the density.
- μ is the dynamic viscosity. Here we have assumed that the fluid is Newtonian such that μ is constant.

Rewritten

$$\nabla \cdot \boldsymbol{u} = 0$$

$$\rho \frac{\partial \boldsymbol{u}}{\partial t} + \rho \left(\boldsymbol{u} \cdot \nabla \right) \boldsymbol{u} = -\nabla p + \mu \nabla^2 \boldsymbol{u} + \rho \boldsymbol{g}$$

Can be rewritten as:

$$\nabla \cdot \boldsymbol{u} = 0$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + \left(\boldsymbol{u} \cdot \nabla \right) \boldsymbol{u} = -\nabla p^* + \nu \nabla^2 \boldsymbol{u} + \boldsymbol{g}$$

- ν is the kinematic viscosity. This is the real viscosity that controls the Reynolds number.
- p^* is the reduced pressure.

Boundary conditions

No-slip

Fluid in contact with an object goes at the velocity of said object.

$$\boldsymbol{u} = \boldsymbol{u}_o$$

This is the traditional boundary condition.

Slip / no-penetration

There is no velocity normal to the object.

$$\boldsymbol{u} \cdot \boldsymbol{n} = 0$$

This is often used for symmetry or free surfaces.

Boundary conditions

Outlets

Outlets are tricky. We can either consider outlets to be:

$$p^* = 0$$

$$\nabla \boldsymbol{u} \cdot \boldsymbol{n} = 0$$

or

$$-p^* + \nu \nabla \boldsymbol{u} \cdot \boldsymbol{n} = 0$$

Both have a different meaning.

What's pressure?

Recall that the incompressible Navier-Stokes equations have the following equation of state:

$$\rho = C$$

Let's try to understand some physical implications of this.

What's pressure?

Recall that the incompressible Navier-Stokes equations have the following equation of state:

$$\rho = C$$

Let's try to understand some physical implications of this. Recall the definition of the speed of sound:

$$c^2 = \left(\frac{\partial p}{\partial \rho}\right)_s$$

What are the consequences for the incompressible Navier-Stokes?

What's pressure?

Recall that the incompressible Navier-Stokes equations have the following equation of state:

$$\rho = C$$

Let's try to understand some physical implications of this. Recall the definition of the speed of sound:

$$c^2 = \left(\frac{\partial p}{\partial \rho}\right)_s$$

What are the consequences for the incompressible Navier-Stokes?

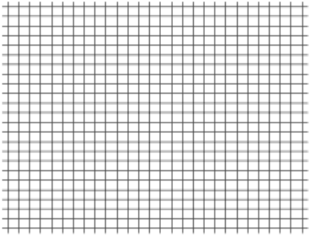
The speed of sound is infinite! There is no sound, information propagates everywhere.

Sound?

What is sound? (Baby don't hurt me...)

In physics, sound is a vibration that propagates as an acoustic wave, through a transmission medium such as a gas, liquid or solid.

- Transmitted through gases and liquids as longitudinal waves.
- Longitudinal sound waves are waves of alternating pressure deviations from the equilibrium pressure, causing local regions of compression and rarefaction.



Measure of sound

Units

Sound is measured in decibels

- decibel (dB), unit for expressing the ratio between two physical quantities. One decibel (0.1 bel) equals 10 times the common logarithm of the power ratio.

Consequence of incompressibility

- Speed of sound is infinite.
- Sound is pressure waves.
- Pressure waves propagate instantanously everywhere.

Measure of sound

Units

Sound is measured in decibels

- decibel (dB), unit for expressing the ratio between two physical quantities. One decibel (0.1 bel) equals 10 times the common logarithm of the power ratio.

Consequence of incompressibility

- Speed of sound is infinite.
- Sound is pressure waves.
- Pressure waves propagate instantanously everywhere.

Measure of sound

Units

Sound is measured in decibels

- decibel (dB), unit for expressing the ratio between two physical quantities. One decibel (0.1 bel) equals 10 times the common logarithm of the power ratio.

Consequence of incompressibility

- Speed of sound is infinite.
- Sound is pressure waves.
- Pressure waves propagate instantanously everywhere.

Measure of sound

Units

Sound is measured in decibels

- decibel (dB), unit for expressing the ratio between two physical quantities. One decibel (0.1 bel) equals 10 times the common logarithm of the power ratio.

Consequence of incompressibility

- Speed of sound is infinite.
- Sound is pressure waves.
- Pressure waves propagate instantanously everywhere.

An equation for pressure

Still, let's try to build an equation for pressure.

$$\nabla \cdot \boldsymbol{u} = 0 \tag{1}$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\, \boldsymbol{u} = -\nabla p^* + \nu \nabla^2 \boldsymbol{u} \tag{2}$$

Pressure equation

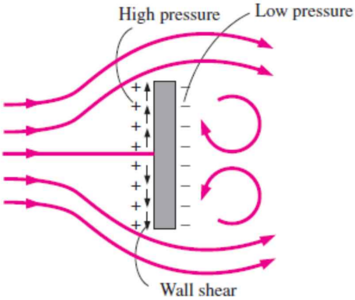
$$\nabla \boldsymbol{u} : \nabla \boldsymbol{u} = -\nabla^2 p^* \tag{3}$$

- Pressure is driven by a Poisson equation which depends on the instantaneous velocity field.
- Pressure has no time derivative. It is never transient.
- Pressure is a Lagrange multiplier. It is a constraint to impose mass conservation.
- Pressure is linked to the continuity equation, not to the momentum conservation equation.
- Generally, it is what couples the velocity components amongst themselves.

Interpretation

$$\nabla \cdot \boldsymbol{u} = 0 \tag{4}$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\, \boldsymbol{u} = -\nabla p^* + \nu \nabla^2 \boldsymbol{u} \tag{5}$$



Conclusion

- Incompressible Navier-Stokes equations imply a constant density.
- Constant density implies an infinite speed of sound.
- Pressure is a Lagrange multiplier which is driven by a Poisson equation.
- This, with turbulence, is what makes the solution of the Navier-Stokes equations so difficult. Now that we understand what we are facing, let's face it together...

Conclusion

- Incompressible Navier-Stokes equations imply a constant density.
- Constant density implies an infinite speed of sound.
- Pressure is a Lagrange multiplier which is driven by a Poisson equation.
- This, with turbulence, is what makes the solution of the Navier-Stokes equations so difficult. Now that we understand what we are facing, let's face it together...

Conclusion

- Incompressible Navier-Stokes equations imply a constant density.
- Constant density implies an infinite speed of sound.
- Pressure is a Lagrange multiplier which is driven by a Poisson equation.
- This, with turbulence, is what makes the solution of the Navier-Stokes equations so difficult. Now that we understand what we are facing, let's face it together...

Conclusion

- Incompressible Navier-Stokes equations imply a constant density.
- Constant density implies an infinite speed of sound.
- Pressure is a Lagrange multiplier which is driven by a Poisson equation.
- This, with turbulence, is what makes the solution of the Navier-Stokes equations so difficult. Now that we understand what we are facing, let's face it together...

Outline

Weak form: Stokes

Let's start with the steady Stokes equation and establish its weak form.

$$\nabla \cdot \boldsymbol{u} = 0 \tag{6}$$

$$\nabla p^* - \nu \nabla^2 \boldsymbol{u} = 0 \tag{7}$$

Notation

- q is the test function for pressure, $\boldsymbol{v}$ the test function for velocity. We have as many test functions $\boldsymbol{v}$ as we have velocity components $\boldsymbol{u}$.
- Since pressure is related to mass conservation, we will use q to test (6) and $\boldsymbol{v}$ to test (7)
- Eq. (7) is a vector equation. So testing it with $\boldsymbol{v}$ will imply a scalar product!
- This will be difficult. Stop me if you have questions. If you do not understand the first time, it just means you are a human being.

Weak form: Stokes

Let's start with the steady Stokes equation and establish its weak form.

$$\nabla \cdot \boldsymbol{u} = 0 \tag{6}$$

$$\nabla p^* - \nu \nabla^2 \boldsymbol{u} = 0 \tag{7}$$

Notation

- q is the test function for pressure, $\boldsymbol{v}$ the test function for velocity. We have as many test functions $\boldsymbol{v}$ as we have velocity components $\boldsymbol{u}$.
- Since pressure is related to mass conservation, we will use q to test (6) and $\boldsymbol{v}$ to test (7)
- Eq. (7) is a vector equation. So testing it with $\boldsymbol{v}$ will imply a scalar product!
- This will be difficult. Stop me if you have questions. If you do not understand the first time, it just means you are a human being.

Weak form: Stokes

Let's start with the steady Stokes equation and establish its weak form.

$$\nabla \cdot \boldsymbol{u} = 0 \tag{6}$$

$$\nabla p^* - \nu \nabla^2 \boldsymbol{u} = 0 \tag{7}$$

Notation

- q is the test function for pressure, $\boldsymbol{v}$ the test function for velocity. We have as many test functions $\boldsymbol{v}$ as we have velocity components $\boldsymbol{u}$.
- Since pressure is related to mass conservation, we will use q to test (6) and $\boldsymbol{v}$ to test (7)
- Eq. (7) is a vector equation. So testing it with $\boldsymbol{v}$ will imply a scalar product!
- This will be difficult. Stop me if you have questions. If you do not understand the first time, it just means you are a human being.

Weak form: Stokes

Let's start with the steady Stokes equation and establish its weak form.

$$\nabla \cdot \boldsymbol{u} = 0 \tag{6}$$

$$\nabla p^* - \nu \nabla^2 \boldsymbol{u} = 0 \tag{7}$$

Notation

- q is the test function for pressure, $\boldsymbol{v}$ the test function for velocity. We have as many test functions $\boldsymbol{v}$ as we have velocity components $\boldsymbol{u}$.
- Since pressure is related to mass conservation, we will use q to test (6) and $\boldsymbol{v}$ to test (7)
- Eq. (7) is a vector equation. So testing it with $\boldsymbol{v}$ will imply a scalar product!
- This will be difficult. Stop me if you have questions. If you do not understand the first time, it just means you are a human being.

Weak form: Stokes

Let's start with the steady Stokes equation and establish its weak form.

$$\nabla \cdot \boldsymbol{u} = 0 \tag{6}$$

$$\nabla p^* - \nu \nabla^2 \boldsymbol{u} = 0 \tag{7}$$

Notation

- q is the test function for pressure, $\boldsymbol{v}$ the test function for velocity. We have as many test functions $\boldsymbol{v}$ as we have velocity components $\boldsymbol{u}$.
- Since pressure is related to mass conservation, we will use q to test (6) and $\boldsymbol{v}$ to test (7)
- Eq. (7) is a vector equation. So testing it with $\boldsymbol{v}$ will imply a scalar product!
- This will be difficult. Stop me if you have questions. If you do not understand the first time, it just means you are a human being.

Weak form: Stokes

$$\iiint_{\Omega} q\nabla \cdot \boldsymbol{u}\mathrm{d}\Omega = 0$$

$$\iiint_{\Omega} -\left(\nabla \cdot \boldsymbol{v}\right) p^* + \nu\nabla\boldsymbol{v}^T : \nabla\boldsymbol{u}\mathrm{d}\Omega + \iint_{\Gamma} \boldsymbol{v} \cdot \left(p^*\boldsymbol{n} - \nu\left(\nabla\boldsymbol{u}\right)^T \cdot \boldsymbol{n}\right)\mathrm{d}\Gamma = 0$$

Our natural boundary condition has changed:

$$\iint_{\Gamma} \boldsymbol{v} \cdot \left(p^*\boldsymbol{n} - \nu\left(\nabla\boldsymbol{u}\right)^T \cdot \boldsymbol{n}\right)\mathrm{d}\Gamma \tag{8}$$

is equivalent to a zero traction boundary condition. This is very similar to a zero pressure outlet.

Weak form: Stokes

Introducing our interpolation polynomial:

- ψ_i is the Lagrange polynomial for pressure.
- $\boldsymbol{\phi}_j$ is the vector Lagrange polynomial for velocity. It is a vector of `dim` Lagrange polynomials.

$$\sum_j \boldsymbol{u}_j \iiint_\Omega \psi_i \nabla \cdot \boldsymbol{\phi}_j \mathrm{d}\Omega = 0$$

$$\sum_k -p_k^* \iiint_\Omega \left(\nabla \cdot \boldsymbol{\phi_i}\right) \psi_k \mathrm{d}\Omega + \sum_j \boldsymbol{u}_j \iiint_\Omega \nu \nabla \boldsymbol{\phi}_i^T : \nabla \boldsymbol{\phi}_j \mathrm{d}\Omega$$
$$+ \sum_k p_k^* \iint_\Gamma \psi_k \boldsymbol{\phi}_i \cdot \boldsymbol{n} \mathrm{d}\Gamma - \sum_j \boldsymbol{u}_j \iint_\Gamma \boldsymbol{\phi}_i \cdot \nu (\nabla \boldsymbol{\phi}_j)^T \cdot \boldsymbol{n} \mathrm{d}\Gamma = 0$$

Matrix structure

$$\sum_j \boldsymbol{u}_j \iiint\limits_{\Omega} \psi_i \nabla \cdot \boldsymbol{\phi}_j \mathrm{d}\Omega = 0$$

$$\sum_k -p_k^* \iiint\limits_{\Omega} \left(\nabla \cdot \boldsymbol{\phi_i}\right) \psi_k \mathrm{d}\Omega + \sum_j \boldsymbol{u}_j \iiint\limits_{\Omega} \nu \nabla \boldsymbol{\phi}_i^T : \nabla \boldsymbol{\phi}_j \mathrm{d}\Omega$$
$$+ \sum_k p_k^* \iint\limits_{\Gamma} \psi_k \boldsymbol{\phi}_i \cdot \boldsymbol{n} \mathrm{d}\Gamma - \sum_j \boldsymbol{u}_j \iint\limits_{\Gamma} \boldsymbol{\phi}_i \cdot \nu (\nabla \boldsymbol{\phi}_j)^T \cdot \boldsymbol{n} \mathrm{d}\Gamma = 0$$

What will be the matrix structure? Let's assume we can write in block form:

$$\mathcal{M} = \begin{bmatrix} A & B^T \\ B & C \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ p^* \end{bmatrix} = 0 \tag{9}$$

Matrix structure: A

$$\mathcal{M} = \begin{bmatrix} A & B^T \\ B & C \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ p^* \end{bmatrix} = 0 \tag{10}$$

The A block is given by all the terms that combine ϕ_i and ϕ_j:

$$A = \iiint_\Omega \nu \nabla \boldsymbol{\phi}_i^T : \nabla \boldsymbol{\phi}_j \mathrm{d}\Omega$$

Matrix structure: B^T

$$\mathcal{M} = \begin{bmatrix} A & B^T \\ B & C \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ p^* \end{bmatrix} = 0 \tag{11}$$

The B^T block is given by all the terms that combine ϕ_i and ψ_k:

$$B^T = \iiint\limits_{\Omega} - \left(\nabla \cdot \boldsymbol{\phi_i}\right) \psi_k \mathrm{d}\Omega$$

Matrix structure: B

$$\mathcal{M} = \begin{bmatrix} A & B^T \\ B & C \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ p^* \end{bmatrix} = 0 \tag{12}$$

The B block is given by all the terms that combine ψ_i and ϕ_j:

$$B = \iiint_\Omega \psi_i \nabla \cdot \boldsymbol{\phi}_j \mathrm{d}\Omega$$

Matrix structure: C

$$\mathcal{M} = \begin{bmatrix} A & B^T \\ B & C \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ p^* \end{bmatrix} = 0 \tag{13}$$

The C block is given by all the terms that combine ψ_i and ψ_k:

$$C = 0$$

Matrix structure: C

$$\mathcal{M} = \begin{bmatrix} A & B^T \\ B & C \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ p^* \end{bmatrix} = 0 \tag{13}$$

The C block is given by all the terms that combine ψ_i and ψ_k:

$$C = 0$$

Wait wut?

Saddle point problem

$$\mathcal{M} = \begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ p^* \end{bmatrix} = 0 \tag{14}$$

The system of equations we have to solve for the Stokes problem leads to a saddle-point problem. These matrices are very difficult to solve because the zero block on the diagonal.

- If we use a direct solver, this will not pose any particular problem.
- For iterative solver, this requires very specific preconditioning techniques. Hence people have developed approaches to circumvent this limitation.

Code complexity

- Our code needs to be able to understand that we now have multiple components per location where we store the degree of freedom. Our test function $\boldsymbol{v}$ is a `Tensor<1,dim>` and our gradient $\nabla \boldsymbol{v}$ is a `Tensor<2,dim>`. These are much more complicated mathematical objects to manipulate.
- At every location where we store degrees of freedom, we now have multiple variables stored. In 2D, we now store u_x, u_y, p instead of just storing T. This generates much more complicated sparsity patterns. This is also significantly complicated to implement.
- Assembling our equations will be more difficult.

Ladyzhenskaya–Babuška–Brezzi condition

But wait, there's more?

The Ladyzhenskaya–Babuška–Brezzi condition is a sufficient condition for a saddle point problem to have a unique solution that depends continuously on the input data. This condition applies to all saddle point problems like the Stokes and the Navier-Stokes equations

What does it mean?

When a variable is a constraint (pressure) over a field (velocity), the solution space for the constraint cannot be equal to or larger than the variable it constraints. Otherwise, we obtain checkerboard effect.

Ladyzhenskaya–Babuška–Brezzi condition

Consequence

The combination of some element type for pressure and some element type for velocity are not LBB compatible. They lead to solutions which will not converge as we refine the mesh.

- Q_n for velocity and Q_n for pressure **are not** LBB stable
- Q_n for velocity and Q_{n-1} for pressure **are** LBB stable

Navier-Stokes

The same process can be applied to the Navier-Stokes equations, but we need to first linearize the problem.

$$\nabla \cdot \boldsymbol{u} = 0 \tag{15}$$

$$(\boldsymbol{u} \cdot \nabla)\, \boldsymbol{u} + \nabla p^* - \nu \nabla^2 \boldsymbol{u} = 0 \tag{16}$$

Weak form: Navier-Stokes

The residual is:

$$R(p) = \nabla \cdot \boldsymbol{u} \tag{17}$$
$$R(\boldsymbol{u}) = (\boldsymbol{u} \cdot \nabla)\, \boldsymbol{u} + \nabla p^* - \nu \nabla^2 \boldsymbol{u} \tag{18}$$

The problem for which we need to formulate the weak form is:

$$\nabla \cdot \boldsymbol{\delta u} = -R(p) \tag{19}$$
$$(\boldsymbol{\delta u} \cdot \nabla)\, \boldsymbol{u} + (\boldsymbol{u} \cdot \nabla)\, \boldsymbol{\delta u} + \nabla \delta p^* - \nu \nabla^2 \boldsymbol{\delta u} = -R(\boldsymbol{u}) \tag{20}$$

There is significant non-linearity which arises from the advection term.

Partial conclusions

Solving the incompressible Navier-Stokes equations lead to the following problems:

- Solve a vector non-linear problem
- Every iteration requires solving a saddle-point problem
- Not all elements respect the LBB condition

This is a general challenge with the Navier-Stokes equations. We will learn two different strategies to circumvent these challenges and one to face them head-on.

Outline

Straightforward solution methods

Straightforward solution method consist in solving the system matrix that arises from the Navier-Stokes equation efficiently:

$$\mathcal{M} = \begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ p^* \end{bmatrix} = 0 \tag{21}$$

The main ideas of the algorithm are:

- Assemble Jacobian matrix and right-hand side
- Formulate adequate preconditioner for the equations
- Solve linear system
- Iterate until residual is zero

The main challenge is in formulating an adequate preconditioner for the matrix. It is not an easy endeavour and remains an active area of research.

Conclusion

Main challenges

The main difficulty in solving the Navier-Stokes lies in:

- Assembling the matrix (can be an expensive operation)
- Formulating an adequate preconditioner to solve the linear system that arises

This are active areas of research.

Other issues remain

Other issues with the solution of advection problem remain. When the Péclet number becomes too high, it remains necessary to introduce stabilization into the scheme (SUPG).

Outline

Understanding components

In the Navier-stokes equations, we now have multiple DoF that can reside at a same location.

- Some must be interpreted as part of a vector ($\boldsymbol{u}$) while some not
- The `DoFHandler` must be aware of this notion
- The `FEValues` must also be aware of this notion

What does this change?

- The FE object becomes an `FESystem` made of multiple FE
- The `DoFHandler` receives the `FESystem` when being initialized
- The components of the `DoFHandler` can be interpreted by `FEValuesExtractors`

The code: Understanding components

This will alter multiple parts of the code. The first one is the declaration of the `fe` and the initialization of the `DoFHandler`

```
// The FeValuesExtractors enable us to interpret components as
    scalars or tensors.
// This will be part of our solver class.
FESystem<dim>   fe;

// Instantiating it will be different also
// as it is built from multiple regular FE_Q
fe(FE_Q<dim>(degree + 1), dim, FE_Q<dim>(degree), 1)
```

The code: Understanding components

```
// The FeValuesExtractors enable us to interpret components as
    scalars or tensors
const FEValuesExtractors::Vector velocities(0);
const FEValuesExtractors::Scalar pressure(dim);

// You can now reconstruct the velocity and the pressure
// from the components
 for (const auto &cell : dof_handler.active_cell_iterators())
{
  fe_values.reinit(cell);
  fe_values[velocities].get_function_values(evaluation_point,
                              present_velocity_values);

  fe_values[velocities].get_function_gradients(
                        evaluation_point, present_velocity_gradients);

  fe_values[pressure].get_function_values(evaluation_point,
        present_pressure_values);
}
```

The code: understanding components

```
const FEValuesExtractors::Vector velocities(0);
const FEValuesExtractors::Scalar pressure(dim);

// Same can be done for shapes
// from the components
for (const auto &cell : dof_handler.active_cell_iterators())
{
   for (unsigned int k = 0; k < dofs_per_cell; ++k)
   {
      // divergence of shape
      fe_values[velocities].divergence(k, q);
      // gradient of shape
      grad_phi_u[k] = fe_values[velocities].gradient(k, q);
      // shape function
      phi_u[k]    = fe_values[velocities].value(k, q);
   }
}
```

Vector test functions

Vector test functions are complicated to understand. To better understand them, let's look at the following code:

```
 // Loop over the cells
for (const auto &cell : dof_handler.active_cell_iterators())
{
  for (unsigned int q = 0; q < n_q_points; ++q)
   // Loop over the degrees of freedom
   for (unsigned int i = 0; i < dofs_per_cell; ++i)
   {
      // shape function
      std::cout << fe_values[velocities].value(k, q) <<std::endl ;
      std::cout << fe_values[pressure].value(k, q) <<std::endl ;
   }
}
```

How many `dofs_per_cell` would I have in 2D Q1-Q1?

Vector test functions

Vector test functions are complicated to understand. To better understand them, let's look at the following code:

```
// Loop over the cells
for (const auto &cell : dof_handler.active_cell_iterators())
{
 for (unsigned int q = 0; q < n_q_points; ++q)
    // Loop over the degrees of freedom
   for (unsigned int i = 0; i < dofs_per_cell; ++i)
   {
    // shape functions
    std::cout << fe_values[velocities].value(k, q) <<std::endl;
    std::cout << fe_values[pressure].value(k, q) <<std::endl;
    }
}
```

What value does my shape function take in Q1-Q1? Q2-Q1?

Result: Q1-Q1

DOF no.	Component	v_x	v_y	q
0	0	X		
1	1		X	
2	2			X
3	0	X		
4	1		X	
5	2			X
6	0	X		
7	1		X	
8	2			X
9	0	X		
10	1		X	
11	2			X

Even though $\boldsymbol{v}$ is a vector, it always ends up being a unit vector. This might seem redundant, but it makes writing the weak form so much simpler.

Result: Q2-Q1

DOF no.	Component	v_x	v_y	q
0	0	X		
1	1		X	
2	2			X
3	0	X		
4	1		X	
5	2			X
6	0	X		
7	1		X	
8	2			X
9	0	X		
10	1		X	
11	2			X
12	0	X		
13	1		X	
14	0	X		
15	1		X	
16	0	X		
17	1		X	
18	0	X		
19	1		X	
20	0	X		
21	1		X	

Outline

Predictor-corrector methods

Premise

The premise behind predictor-corrector methods is to simplify the system of equations that arises by separating the momentum equation from the pressure equation and building an equation for pressure which is easier to solve.

Concept

The predictor-corrector methods allow us to separate the solution of momentum and pressure leading to:

- A smaller system of equations.
- No saddle-point problem.

But for this, we need an equation for pressure.

Equation for pressure

$$\nabla \cdot \boldsymbol{u} = 0 \tag{22}$$

$$(\boldsymbol{u} \cdot \nabla)\,\boldsymbol{u} + \nabla p^* - \nu \nabla^2 \boldsymbol{u} = 0 \tag{23}$$

In Einstein's notation:

$$\partial_i u_i = 0 \tag{24}$$

$$u_j \partial_j u_i + \partial_i p^* - \nu \partial_j \partial_j u_i = 0 \tag{25}$$

Let's reconstruct an equation for pressure.

Equation for pressure

$$u_j\partial_j u_i + \partial_i p^* - \nu\partial_j\partial_j u_i = 0 \tag{26}$$
$$\partial_i\partial_i p = -\partial_i(u_j\partial_j u_i) \tag{27}$$

or

$$\boldsymbol{u}\cdot\nabla\boldsymbol{u} + \nabla p^* - \nu\nabla^2 u_i = 0 \tag{28}$$
$$\nabla^2 p^* = -\nabla\cdot((\boldsymbol{u}\cdot\nabla)\boldsymbol{u}) \tag{29}$$

Introducing this additional equation has induced an additional boundary condition on Γ:

$$\nabla p = 0 \tag{30}$$

The idea behind predictor corrector methods is that you can solve them subsequently.

- Solve momentum
- Solve pressure

and that at convergence, you will obtain the desired flow.

Predictor-Corrector methods

Predictor-corrector methods work by introducing a Poisson equation for pressure:

Advantages

- Compatible with Q_n/Q_n elements.
- Easier to precondition the matrices (and they are smaller).
- Boundary condition for pressure remain an active area of discussion.

Disadvantages

- Convergence can be slow (however, both systems can be put in the same matrix).
- Confusing literature.

Outline

Stabilized methods

Stabilized methods aim at modifying the Navier-Stokes equations to facilitate their solution instead of splitting them into multiple equations. They work in a similar fashion as SUPG method by adding blocks which depend on the strong form of the residual. An example is PSPG/SUPG stabilization:

$$\int_{\Omega} \nabla \cdot \boldsymbol{u} q d\Omega + \sum_{K} \int_{\Omega_k} \left(\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} + \nabla p - \nabla \cdot \boldsymbol{\tau} - \boldsymbol{f} \right) \cdot \left(\tau_u \nabla q \right) d\Omega_k = 0 \tag{31}$$

$$\begin{aligned} &\int_{\Omega} \left(\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} - \boldsymbol{f} \right) \cdot \boldsymbol{v} d\Omega + \int_{\Omega} \boldsymbol{\tau} : \nabla \boldsymbol{v} d\Omega - \int_{\Omega} p \nabla \cdot \boldsymbol{v} d\Omega \\ &+ \sum_{K} \int_{\Omega_k} \left(\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} + \nabla p - \nabla \cdot \boldsymbol{\tau} - \boldsymbol{f} \right) \cdot \left(\tau_u \boldsymbol{u} \cdot \nabla \boldsymbol{v} \right) d\Omega_k = 0 \end{aligned} \tag{32}$$

Stabilized methods

Stabilized methods modify the Navier-Stokes equations:

Advantages

- Compatible with Q_n/Q_n elements.
- Can also serve as turbulence models (Implicit LES).
- Numerical parameters vanish.

Disadvantages

- Difficult literature.
- Lead to large matrices which require careful preconditioning.

Outline

What to do with your solution?

It is not trivial to post-process the result that arise from the Navier-Stokes equations. Here we aim to show some ways the results can be post-processed.

- Streamlines
- Derived fields
- Forces on objects

Streamlines

Streamlines are an interesting way to post-process a velocity field. In essence, they show the trajectories of particles if the velocity field was frozen (e.g. assuming $\partial_t \boldsymbol{u} = 0$). They are obtained by solving the following equation for tracer particles:

$$\partial_t \boldsymbol{x} = \boldsymbol{u}(\boldsymbol{x}) \tag{33}$$

This equation is generally solved using a Runge-Kutta scheme. The challenge here is to adequately interpolate the velocity field to the position of the particles, something that is easily done in FEM.

Derived fields: Vorticity

Multiple fields can be calculated from the velocity and the pressure field. These fields can be useful to help a user understand or postprocess information. A first example is the vorticity:

$$\boldsymbol{\omega} = \nabla \times \boldsymbol{u} \tag{34}$$

The vorticity is especially useful to identify vortices in 2D flows.

Derived fields: Q criterion

The Q criterion is the second invariant of the velocity gradient tensor. It is computed as:

$$Q = \frac{1}{2}\left(\|\Omega\|^2 - \|\mathcal{S}\|^2\right) \tag{35}$$

where $S_{ij} = \frac{1}{2}\left(J_{ij} + J_{ji}\right)$ is the symmetric part of the tensor and $S_{ij} = \frac{1}{2}\left(J_{ij} - J_{ji}\right)$ the anti-symmetric one with $J_{ij} = \partial_j u_i$. Positive value of the Q criterion, along with negative pressure (with respect to the average pressure) indicate vortices.

Forces on object

Using the velocity field and the shape functions, the stress acting on faces can be straightforwardly calculated. To obtain the total force acting on Γ_b, a subset of Γ, the total stress tensor needs to be integrated:

$$\boldsymbol{f}_b = \int_{\Gamma_b} \sigma \cdot \boldsymbol{n} \mathrm{d}\Gamma_b \tag{36}$$

with $\sigma = -p\mathcal{I} + (\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T)$ the stress tensor.

Outline

Conclusions

Challenging

Solving the incompressible Navier-Stokes equations is very difficult. There are many ways to solve them. We have seen three families of strategies:

- Straightforward solution using adequate preconditioning.
- Predict-corrector approaches.
- Stabilized approaches.

Active area of research

This remains an active area of research. In the end, there are many implementation subtleties that differentiate all software.

Homework

The homework will guide you through an existing code that solves the incompressible Navier-Stokes equations in a straightforward fashion.