FEniCS Course

Lecture 9: Incompressible Navier–Stokes

Contributors
Kent-Andre Mardal

The incompressible Navier–Stokes equations

$$
\begin{aligned}
\rho\left(\frac{\partial u}{\partial t} + u \cdot \nabla u\right) &= -\nabla p + \mu \nabla^2 u + f && \text{in } \Omega \times (0, T] \\
\nabla \cdot u &= 0 && \text{in } \Omega \times (0, T] \\
u &= g_{\mathrm{D}} && \text{on } \Gamma_{\mathrm{D}} \times (0, T] \\
\mu \frac{\partial u}{\partial n} - pn &= t_{\mathrm{N}} && \text{on } \Gamma_{\mathrm{N}} \times (0, T] \\
u(\cdot, 0) &= u_0 && \text{in } \Omega
\end{aligned}
$$

- $u : \Omega \to \mathbb{R}^d$ is the unknown fluid velocity
- $p : \Omega \to \mathbb{R}$ is the unknown pressure
- ρ is the fluid density
- μ is the fluid density
- f is a given body force per unit volume
- g_{D} is a given boundary velocity (Dirichlet conditions)
- t_{N} is a given boundary traction (Neumann conditions)
- u_0 is a given initial velocity

The equations are a mix of various equations and are associated with a number of numerical problems!

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

$$\nabla \cdot u = 0$$

- An elliptic term: $\mu \nabla^2 u$
- A parabolic term: $\rho \frac{\partial u}{\partial t} - \mu \nabla^2 u$
- A convection-diffusion term: $\rho u \cdot \nabla u - \mu \nabla^2 u$
- A Stokes problem:$-\mu \nabla^2 u + \nabla p = f; \nabla \cdot u = 0$
- A hyperbolic term: $\rho(\frac{\partial u}{\partial t} + u \cdot \nabla u)$

What techniques should we use?

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

$$\nabla \cdot u = 0$$

- $\mu \nabla^2 u$: use continuous elements, discontinous require care
- $\rho \frac{\partial u}{\partial t} - \mu \nabla^2 u$: use $L-$ stable schemes
- $\rho u \cdot \nabla u - \mu \nabla^2 u$: if μ is small then stabilize
- $-\mu \nabla^2 u + \nabla p = f; \nabla \cdot u = 0$: use elements of higher order for u of than p
- $\rho(\frac{\partial u}{\partial t} + u \cdot \nabla u)$: use $A-$ (not $L-$) stable schemes, discontinous elements!

There is a jungle of solution methods

- For time-dependent flow simulations in 3D, tricks are needed for efficiency!
- We need to discretize in *both* time and space
- First we consider methods where we:
  - Discretize in *time*
  - Do some tricks involving splitting the unknowns
  - Discretize in *space*

  Second we consider methods where we:
  - Discretize in *space*
  - Discretize in *time*
  - Attempt to do some tricks

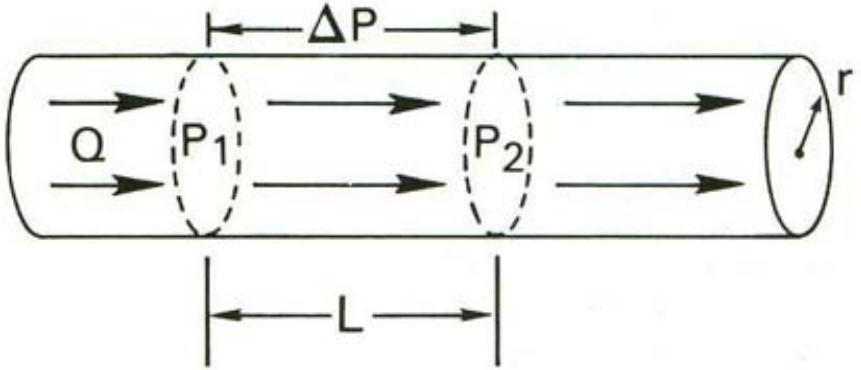
The first approach here is the most efficient, but is associated with problematic boundary conditions and boundary layers, as we will see

The second approach is the most mathematically correct approach, but currently these schemes are typically less efficient than the first approach

Suggested readings

- These notes build on "Langtangen HP, Mardal KA, Winther R. Numerical methods for incompressible viscous flow. Advances in Water Resources. 2002 Dec 31;25(8):1125-46"
- Implementation in FEniCS: "Valen-Sendstad K, Logg A, Mardal KA, Narayanan H, Mortensen M. A comparison of finite element schemes for the incompressible Navier–Stokes equations. In: Automated Solution of Differential Equations by the Finite Element Method 2012 (pp. 399-420). Springer Berlin Heidelberg"
- More comprehensive material: Gresho PM, Sani RL. Incompressible flow and the finite element method, two volume book.

The famous Poiseuille flow



POISEUILLE'S LAW

$$Q = \frac{\Delta P \; r^4 \; \pi}{\eta L \quad 8}$$

Poiseuille flow with $P_2 - P_1$ elements

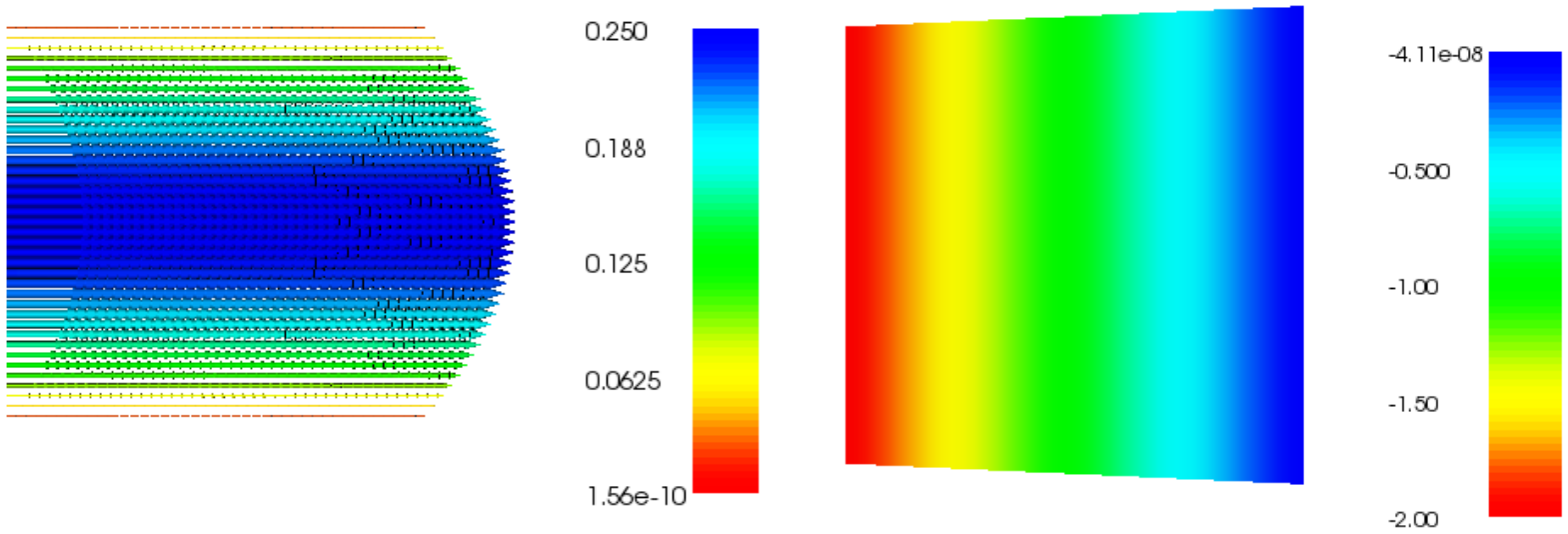


Figure: Illustration of Poiseuille flow in 2D as computed with $P_2 - P_1$ elements in FEniCS. Left image shows the velocity vectors while the right image shows the pressure. Both velocity and pressure are correct up to round-off error.

Poiseuille flow with $P_1 - P_1$ elements

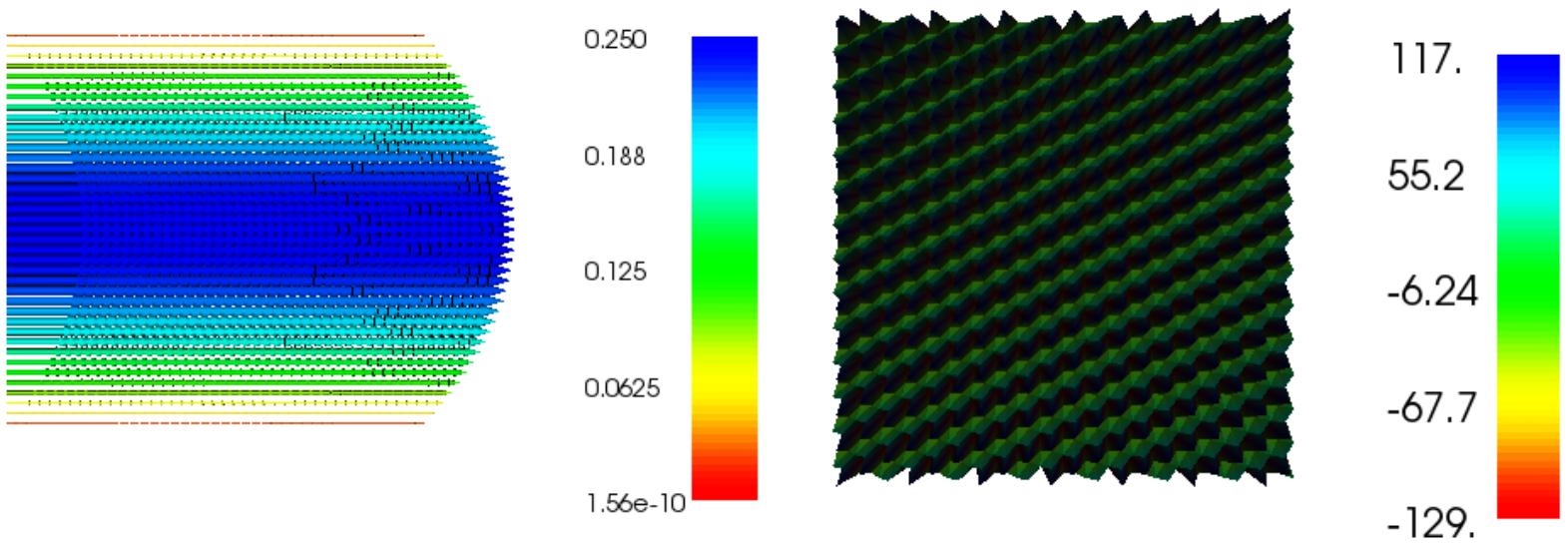
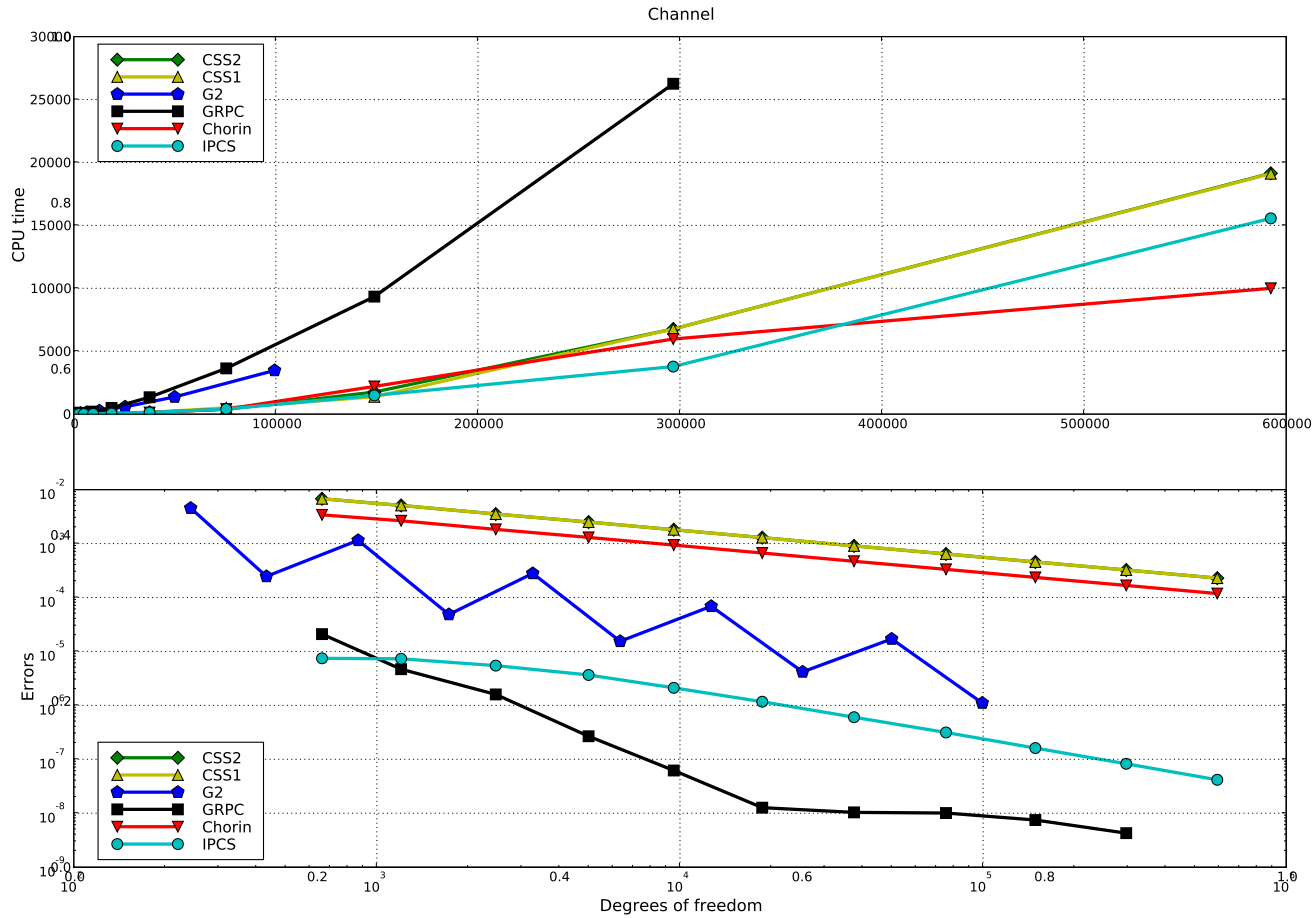


Figure: Illustration of Poiseuille flow in 2D as computed with $P_1 - P_1$ elements in FEniCS. Left image shows the velocity vectors while the right image shows the pressure. The velocity is correct but the pressure is *not*. The $P_1 - P_1$ discretization violates the inf-sup condition.

Efficiency and Accuracy for Channel flow, 2D



Very different behaviour of the different solvers. The methods we discuss here are closely related to IPCS and GRPC.

Explicit scheme, discretize *time* before *space*

The first equation:

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

Here $\nu = \mu/\rho$. When using a explicit Euler becomes

$$u^{n+1} = u^n + \Delta t(-u^n \cdot \nabla u^n - \frac{1}{\rho}\nabla p^n + \nu \nabla^2 u^n + f^n)$$

Two main problems:

- No updated for the pressure (p^{n+1} does not show up anywhere)
- No reason $\nabla \cdot u^{n+1}$ should be zero

Derivation of project schemes

Lets say that our computed u^{n+1} is a tentative guess, u^*

$$u^* = u^n + \Delta t(-u^n \cdot \nabla u^n) - \frac{1}{\rho}\nabla p^n + \nu\nabla^2 u^n + f^n$$

Then the real u^{n+1} should satisfy

$$u^{n+1} = u^n + \Delta t(-u^n \cdot \nabla u^n - \frac{1}{\rho}\nabla p^{n+1} + \nu\nabla^2 u^n + f^n)$$

In otherwords, subtracting the first from the second equation we obtain

$$u^{n+1} - u^* = -\frac{\Delta t}{\rho}\nabla(p^{n+1} - p^n)$$

Derivation of project schemes, cont'd

let

$$u^c = u^{n+1} - u^*$$

and

$$\phi = (p^{n+1} - p^n)$$

then

$$u^c - \frac{\Delta t}{\rho}\nabla\phi = 0 \tag{1}$$

Further, since

$$\nabla \cdot u^{n+1} = 0$$

we obtain

$$\nabla \cdot u^c = -\nabla \cdot u^* \tag{2}$$

Inserting (1) into (2) we obtain

$$-\frac{\Delta t}{\rho}\nabla^2\phi = -\nabla \cdot u^*$$

We note that

$$u^{n+1} = u^* - u^c = u^* - \frac{\Delta t}{\rho}\nabla\phi$$

Derivation of project schemes, cont'd

Putting it all together we have 4 *easy* steps:

1. Compute the tentative velocity

$$u^* = u^n + \Delta t(-u^n \cdot \nabla u^n) - \frac{1}{\rho}\nabla p^n + \nu \Delta u^n + f^n$$

2. Solve a Poisson problem for the pressure update

$$-\nabla^2 \phi = -\frac{\rho}{\Delta t}\nabla \cdot u^*$$

3. Update velocity

$$u^{n+1} = u^* - \frac{\Delta t}{\rho}\nabla \phi$$

4. Update pressure

$$p^{n+1} = p^n + \phi$$

Derivation of project schemes, cont'd

- We developed a scheme involving only a Poisson equation and explicit updates!
- We skipped details concerning the boundary conditions, we will look into that later
- This method of thinking, where we first construct a tentative velocity and then modify to account for the divergence constraint and update the pressure is a general strategy!
- We now turn to implicit methods, but remember that this explicit scheme is very easy to implement on parallel computers

An implicit projection scheme

A tentative guess, involving an explicit pressure

$$u^* - \Delta t((-u^* \cdot \nabla u^*) - \frac{1}{\rho}\nabla p^n + \nu \nabla^2 u^*) = u^n + \Delta t f^{n+1}$$

This results in a nonlinear problem which is seldom used. Let us therefore linearize to obtain

$$u^* - \Delta t((-u^n \cdot \nabla u^*) - \frac{1}{\rho}\nabla p^n + \nu \nabla^2 u^*) = u^n + \Delta t f^{n+1} \quad (3)$$

and what we really want is slightly different

$$u^{n+1} - \Delta t((-u^n \cdot \nabla u^{n+1}) - \frac{1}{\rho}\nabla p^{n+1} + \nu \nabla^2 u^{n+1}) = u^n + \Delta t f^{n+1} \quad (4)$$

Letting

$$u^{n+1} = u^* + u^c$$

and subtracting (4) from (3) we obtain a more complicated expression

An implicit projection scheme, cont'd

Let us, to simplify notation, introduce a convection-diffusion operator s:

$$s(u^c) = \Delta t(-u^n \cdot \nabla u^c + \nu \nabla^2 u^c)$$

Then the velocity correction step can be written as

$$\begin{aligned} u^c - s(u^c) + \frac{\Delta t}{\rho} \nabla \phi &= 0 \\ \nabla \cdot u^c &= -\nabla \cdot u^* \end{aligned}$$

What has been gained?

An implicit projection scheme, cont'd

Let us, to simplify notation, introduce a convection-diffusion operator s:

$$s(u^c) = \Delta t(-u^n \cdot \nabla u^c + \nu \nabla^2 u^c)$$

Then the velocity correction step can be written as

$$u^c - s(u^c) + \frac{\Delta t}{\rho} \nabla \phi = 0$$
$$\nabla \cdot u^c = -\nabla \cdot u^*$$

Nothing has been gained!

This problem is just as hard as the original equations!

However, we remark that the $s(u^c)$ term is a first-order term in time.

Hence, we may ignore this term and still obtain a first order approximation. We obtain:

$$u^c + \frac{\Delta t}{\rho} \nabla \phi = 0$$
$$\nabla \cdot u^c = -\nabla \cdot u^*$$

An implicit projection scheme, cont'd

As for the explicit scheme, the equation

$$\begin{aligned} u^c + \frac{\Delta t}{\rho}\nabla\phi &= 0 \\ \nabla \cdot u^c &= -\nabla \cdot u^* \end{aligned}$$

may be written as

$$-\nabla^2\phi = -\frac{\rho}{\Delta t}\nabla \cdot u^*$$

An implicit projection scheme, cont'd

- We have arrived at two sub-problems that are possible to solve efficiently and accurately with current methods
- The scheme is first order, second order schemes have been claimed but (as far as I know) there is always some assumptions that are not always explicitly stated
- We will not go into details on these assumptions (details are in the paper of Langtangen et. al.
- There are issues with this scheme, regarding the boundary conditions
- Lets first summarize the scheme, which has many names, but we call it *incremental pressure correction scheme* (IPCS)

An implicit projection scheme, cont'd

Putting it all together we have 4 steps:

1. Compute the tentative velocity

$$u^* - s(u^*) + \frac{\Delta t}{\rho}\nabla p^n \quad = \quad f^{n+1}$$

2. Solve a Poisson problem for the pressure update

$$-\nabla^2 \phi \quad = \quad -\frac{\rho}{\Delta t}\nabla \cdot u^*$$

3. Update velocity

$$u^{n+1} = u^* - \frac{\Delta t}{\rho}\nabla \phi$$

4. Update pressure

$$p^{n+1} = p^n + \phi$$

Boundary conditions and Splitting/Projection schemes

- Splitting/Projection schemes "always" introduce trouble near the boundaries
- Navier-Stokes (in 3D) requires 3 conditions in every point at the boundary
- IPCS (and all other projection schemes) requires 4 conditions at the boundary (3 for the tentative velocity, 1 for the Poisson equation)
- We remark that Neumann conditions (for Navier-Stokes equations) are often refered to as pressure conditions, but the Neumann condition involve both velocity and pressure, $\mu \frac{\partial u}{\partial n} - pn = t_{\mathrm{N}}$
- Usually, from a physics point of view, the pressure dominates

Boundary conditions cont'd

We may derive boundary conditions for ϕ in two ways:

- From the scheme:

$$u^{n+1} = u^* - \frac{\Delta t}{\rho}\nabla\phi$$

  Since u^{n+1} and u^* have the same boundary conditions, we obtain homogenous Neumann conditions for ϕ, i.e.,

$$\nabla\phi \cdot n = \frac{\rho}{\Delta t}(u^{n+1} - u^*) \cdot n = 0$$

- From the Navier-Stokes equations we have that :

$$\nabla p^n = -\rho(\frac{\partial u^n}{\partial t} + u^n \cdot \nabla u^n) + \mu\nabla^2 u^n + f$$

  Since $\phi = p^{n+1} - p^n$ and $p^{n+1} \neq p^n$ we obtain a non-homogenous condition.

In conclusion: We arrive at two different conditions which both seem perfectly reasonable. The difference is first order.

Boundary conditions cont'd

Let us then use homogenous Neumann conditions. What happens with the velocity when using these conditions?

We have the velocity update

$$u^{n+1} = u^* - \frac{\Delta t}{\rho} \nabla \phi$$

since we use homogenous Neumann for ϕ, $\frac{\partial \phi}{\partial n} = 0$ and

$$u^{n+1} \cdot n = u^* \cdot n$$

and the normal component of u^{n+1} is correct since u^* has the right boundary conditions

However, the tangential component of $\nabla \phi$ is in general non-zero. Therefore, $u^{n+1} \cdot t$ will not be correct, since

$$u^{n+1} \cdot t = u^* \cdot t - \frac{\Delta t}{\rho} \nabla \phi \cdot t$$

Summary, projection schemes

- Projection schemes like IPCS are the most efficient on the market today for transient flows
- The schemes do not require special mixed elements and have for this reason been prefered
- There are issues with boundary layers, errors etc that you need to be aware of
- In many applications, the main interest is what happens on the boundary, if so be careful
- In the scheme, we first compute a tentative velocity, then project this velocity onto the space of divergence free functions
- Is the velocity divergence free after the projection step?

Exercise

- Implement a IPCS scheme to solve the Poiseuille flow problem (in 2D).
- Validate against the Channel flow test (page 407) in the FEniCS book
- Is the velocity divergence free?
- Is there a difference between P2-P1 elements and P1-P1 elements?

Discretization in space before time

A weak form of

$$\rho(\frac{\partial u}{\partial t} + u \cdot \nabla u) = -\nabla p + \mu \nabla^2 u + f \quad \text{in } \Omega \times (0, T] \qquad (5)$$

$$\nabla \cdot u = 0 \quad \text{in } \Omega \times (0, T] \qquad (6)$$

is obtained by multiply the momentum equation, (5), by a test function v and integrate by parts:

$$\int_\Omega \rho(\frac{\partial u}{\partial t} + u \cdot \nabla u) \cdot v \, \mathrm{d}x + \int_\Omega -p \nabla \cdot v + \mu \nabla u : \nabla v \, \mathrm{d}x = \int_\Omega f \cdot v \, \mathrm{d}x + \int_{\Gamma_\mathrm{N}} t_\mathrm{N} \cdot v \, \mathrm{d}$$

Short-hand notation: $\langle \cdot, \cdot \rangle$ is L^2-inner product

$$\langle \rho \frac{\partial u}{\partial t}, v \rangle + \langle \rho u \cdot \nabla u, v \rangle - \langle p, \nabla \cdot v \rangle + \langle \nabla u, \nabla v \rangle = \langle f, v \rangle + \langle t_\mathrm{N}, v \rangle_{\Gamma_\mathrm{N}}$$

Multiply the continuity equation, (6), by a test function q and obtain

$$\langle \nabla \cdot u, q \rangle = 0$$

Discretization in space before time, written in terms of matrices and vectors

We may write the discrete system as a system of differential-algebraic equations (DAEs):

$$\begin{aligned} M\frac{\partial u}{\partial t} + N(u)u + Au + Bp &= f \\ B^T u &= 0 \end{aligned}$$

Here, if N_i are the basis functions of the velocity and L_j are the basis functions of the pressure the

- M is the mass matrix, $M_{ij} = \langle \rho N_i, N_j \rangle$
- A is the stiffness matrix, $A_{ij} = \langle \mu \nabla N_i, \nabla N_j \rangle$
- N is the advection matrix, $N_{ij} = \langle \rho u \cdot \nabla N_i, N_j \rangle$
- B is the discrete gradient matrix, $B_{ij} = \langle N_i, \nabla L_j \rangle$
- B is the discrete divergence matrix, $B^T_{ij} = \langle \nabla \cdot N_i, L_j \rangle$

Discretization in space before time, cont'd

The DAE-system:

$$\begin{aligned} M\frac{\partial u}{\partial t} + N(u)u + Au + Bp &= f \\ B^T u &= 0 \end{aligned}$$

is terrible! It is

- non-linear
- non-symmetric
- indefinite

Efficienct algorithms for solving this system is still and open question. You may get multilevel, domain decomposition methods to work but it is hard and application-dependent

Discretization in space before time, cont'd

- When we did discretization in time before space we derived a scheme consisting of computing a tenative velocity and then a projection step.
- The scheme had two disadvantages: it was first order and there were errors associated with the boundaries
- Both of these disadvantages can in principle be removed by the technique described in the following, but currently at the expense of efficiency

An algebraic splitting scheme

Let us as before first consider an explicit scheme

$$Mu^{n+1} = Mu^n - \Delta t(N(u^n)u^n + Au^n + Bp^n - f^n)$$

As before, we have no way of updating p^{n+1} and the divergence constraint $B^T u^{n+1} = 0$ is not satisifed

Therefore, we follow the previous procedure of computing a tentative guess

$$Mu^* = Mu^n - \Delta t(N(u^n)u^n + Au^n + Bp^n - f^n) \qquad (7)$$

whereas u^{n+1} should satisfy

$$Mu^{n+1} = Mu^n - \Delta t(N(u^n)u^n + Au^n + Bp^{n+1} - f^n) \qquad (8)$$

Subtracting (7) from (8) we obtain

$$M(u^{n+1} - u^*) = -\Delta t B(p^{n+1} - p^n)$$

An algebraic splitting scheme

In addition to

$$M(u^{n+1} - u^*) = -\Delta t B(p^{n+1} - p^n) \tag{9}$$

using previous notation with u^c and ϕ where we have

$$u^{n+1} = u^* + u^c$$

and

$$\phi = p^{n+1} - p^n$$

Since

$$B^T u^{n+1} = 0$$

we must have that

$$B^T u^c = -B^T u^* \tag{10}$$

and insering (9) into (10) we obtain

$$B^T M^{-1} B\phi = -\frac{1}{\Delta t} B^T u^*$$

Summary of explicit algebraic splitting scheme

The scheme consists of 4 steps

1. Compute a tentative velocity

$$Mu^* = Mu^n - \Delta t(N(u^n)u^n + Au^n + Bp^n - f^n)$$

2. Solve an equation for ϕ

$$B^T M^{-1} B\phi = -\frac{1}{\Delta t} B^T u^*$$

3. Compute velocity update

$$u^{n+1} = u^* - \Delta t M^{-1} B\phi$$

4. Compute pressure update

$$p^{n+1} = p^n + \phi$$

Difference between previous projection scheme and the current algebraic scheme

Previously, we solved the following equation for ϕ

$$\nabla^2 \phi = -\frac{1}{\Delta t} \nabla \cdot u^*$$

now the equation reads:

$$B^T M^{-1} B \phi = -\frac{1}{\Delta t} B^T u^*$$

How do these two equations compare?

Difference between previous projection scheme and the current algebraic scheme

Previously, we solved the following equation for ϕ

$$\nabla^2 \phi = -\frac{1}{\Delta t} \nabla \cdot u^*$$

now the equation reads:

$$B^T M^{-1} B \phi = -\frac{1}{\Delta t} B^T u^*$$

- B^T is a discrete divergence
- B is a discrete gradient
- M (and its inverse) is a discrete identity operator

Hence the two equations seem very similar. An important difference is however that ∇^2 requires a set of boundary conditions whereas $B^T M^{-1} B$ have boundary conditions built in.

An implicit algebraic splitting scheme

The previous scheme is straightforward to generalize into an implicit scheme

1. Compute a tentative velocity

$$Mu^* + \Delta t(N(u^n)u^* + Au^* + Bp^n) = Mu^n + \Delta t f^{n+1}$$

2. Solve an equation for ϕ

$$B^T(M + \Delta t(N(u^n) + A))^{-1}B\phi = -\frac{1}{\Delta t}B^T u^*$$

3. Compute velocity update

$$u^{n+1} = u^* - \Delta t(M + \Delta t(N(u^n) + A))^{-1}B\phi$$

4. Compute pressure update

$$p^{n+1} = p^n + \phi$$

The advantages and challenges with the implicit algebraic splitting scheme

The implicit splitting scheme requires us to solve

$$B^T(M + \Delta t(N(u^n) + A))^{-1}B$$

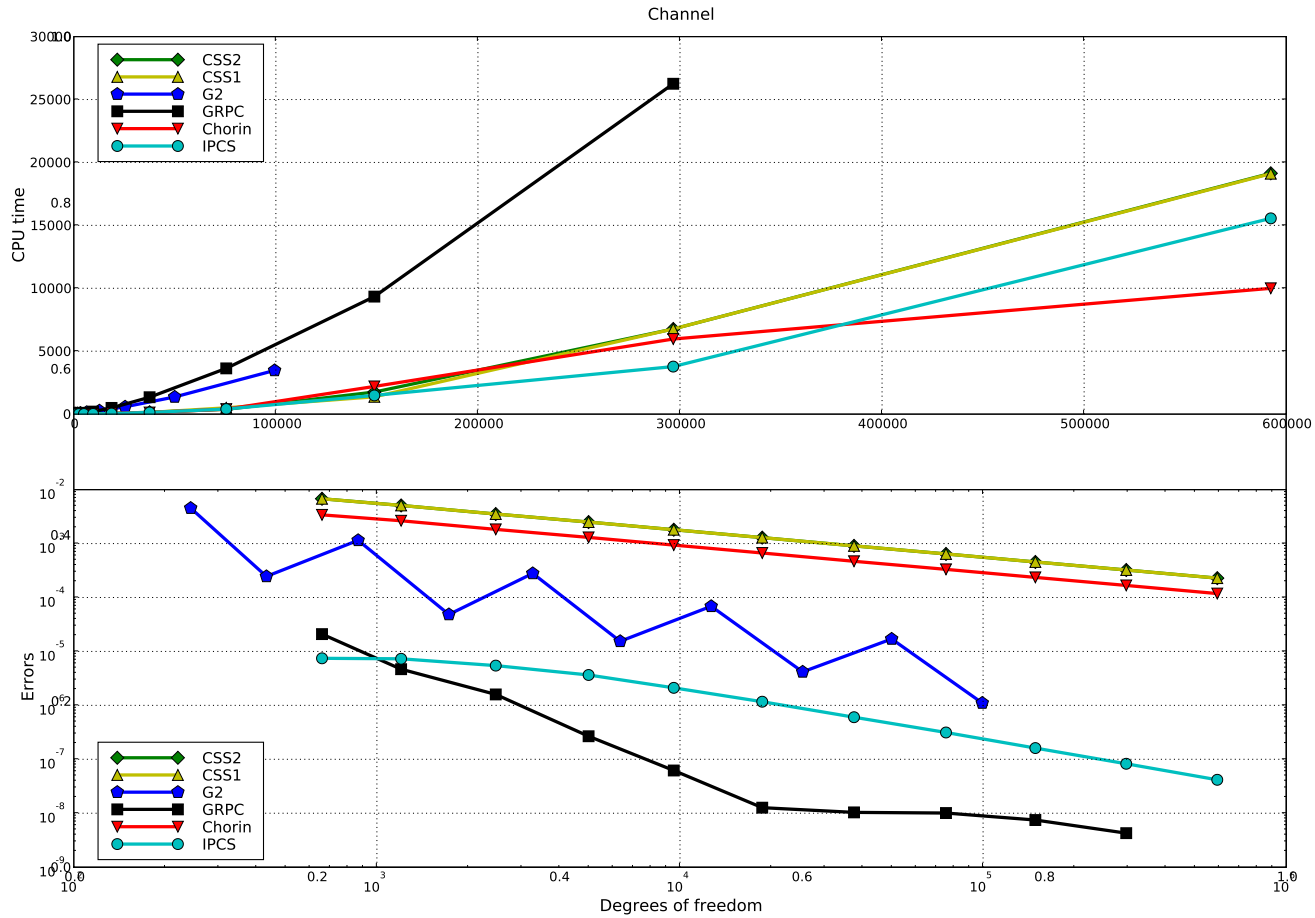
This matrix is usually called the pressure Schur complement

Currently there are no robust and efficient way of solving this even though there are many solution algorithms that work in special cases

And advantage was that there is no need for artificial boundary conditions

Another advantage is that the previous scheme may be ran several times for one time step to obtain a scheme that is more than first order in time (this is in fact utilized in the more method Generalized Richardson iteration for the pressure Schur complement (GRPC) which does precisely this - see the background material)

Efficiency and Accuracy for Channel flow, 2D



Very different behaviour of the different solvers. The methods we discuss here are closely related to IPCS and GRPC.