Computational Fluid Dynamics

Lecture 5
January 30, 2017

Grétar Tryggvason

A Code for the Navier-Stokes Equations in Velocity/Pressure Form

Objectives:

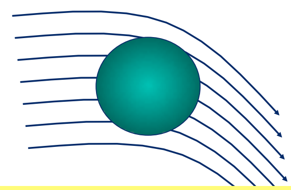
Develop a method to solve the Navier-Stokes equations using "primitive" variables (pressure and velocities), using a control volume approach on a staggered grid.

- Equations
- Discrete Form
- Solution Strategy
- Boundary Conditions
- Code and Results

"Two-dimensional" flow

$$\boldsymbol{u} = (u, v)$$

$$\boldsymbol{n} = (n_x, n_y)$$



Conservation of Momentum

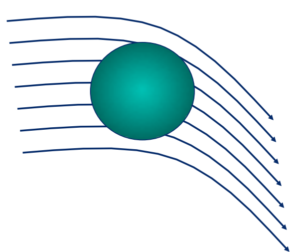
$$\frac{\partial}{\partial t}\int_V u\, dV = -\oint_S u\boldsymbol{u}\cdot\boldsymbol{n}dS - \tfrac{1}{\rho}\oint_S p n_x dS + \nu\oint_S \nabla u\cdot\boldsymbol{n}dS$$

$$\frac{\partial}{\partial t}\int_V v\, dV = -\oint_S v\boldsymbol{u}\cdot\boldsymbol{n}dS - \tfrac{1}{\rho}\oint_S p n_y dS + \nu\oint_S \nabla v\cdot\boldsymbol{n}dS$$

"Two-dimensional" flow

$$\boldsymbol{u} = (u, v)$$

$$\boldsymbol{n} = (n_x, n_y)$$



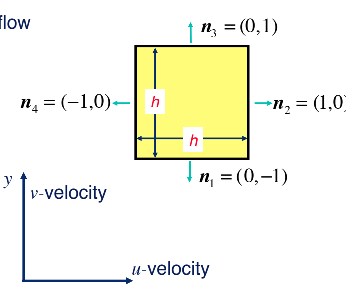
Conservation of Mass

$$\oint_S \boldsymbol{u}\cdot\boldsymbol{n}\, ds = 0$$

Control volume

"Two-dimensional" flow

Select a square control volume, aligned with the coordinate directions



Computational Fluid Dynamics

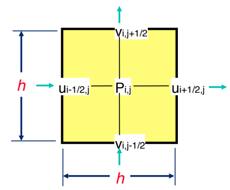
Conservation of mass

Computational Fluid Dynamics

Conservation of mass

$$\oint_S \boldsymbol{u} \cdot \boldsymbol{n} \, ds = 0$$

The velocity at the end of each time step must satisfy this constraint



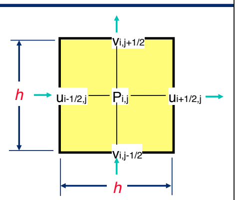
Integrate over the boundary

$$hu^{n+1}_{i+1/2,j} - hu^{n+1}_{i-1/2,j} + hv^{n+1}_{i,j+1/2} - hv^{n+1}_{i,j-1/2} = 0$$

Divide by h: $u^{n+1}_{i+1/2,j} - u^{n+1}_{i-1/2,j} + v^{n+1}_{i,j+1/2} - v^{n+1}_{i,j-1/2} = 0$

Computational Fluid Dynamics

Notice that when we apply the mass conservation equation to a control volume centered at i,j, we naturally pick up the velocities at the edges of the control volume. Nothing has been said so far about how the velocities at the edges are found. They could be interpolated from values at the cell center, or found directly using control volumes centered around the velocity at the edges. The second approach leads to STAGGERED GRIDS

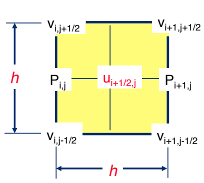


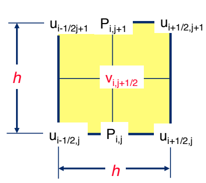
Computational Fluid Dynamics

Conservation of Momentum
The Advection Terms

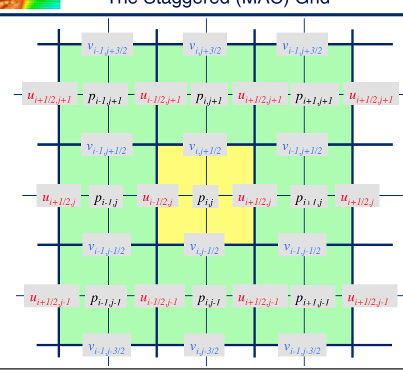
Computational Fluid Dynamics
Momentum equation

For the momentum, select a control volume around each velocity component





Computational Fluid Dynamics
The Staggered (MAC) Grid



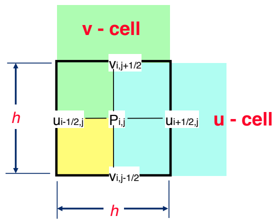
Computational Fluid Dynamics

Define Cell-Averages:

$$u = \frac{1}{V}\int_{V_u} u\,dV$$

$$v = \frac{1}{V}\int_{V_v} v\,dV$$

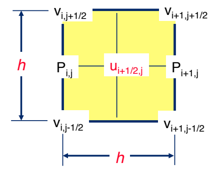
$$P = \frac{1}{V}\int_{V_p} p\,dV$$



Computational Fluid Dynamics
Unsteady term

Rate of change of x-momentum

$$\frac{\partial}{\partial t}\int_V u\,dV$$



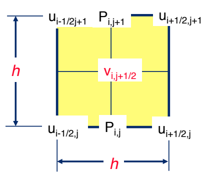
Integrate over the control volume

$$\frac{\partial}{\partial t}\int_V u\,dV \approx \frac{u^{n+1}_{i+1/2,j} - u^n_{i+1/2,j}}{\Delta t} h^2$$

Computational Fluid Dynamics
Unsteady term

Rate of change of y-momentum

$$\frac{\partial}{\partial t}\int_V v\,dV$$



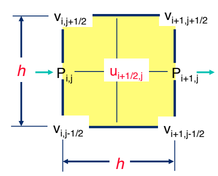
Integrate over the control volume

$$\frac{\partial}{\partial t}\int_V v\,dV \approx \frac{v^{n+1}_{i,j+1/2} - v^n_{i,j+1/2}}{\Delta t} h^2$$

Computational Fluid Dynamics
Advection terms

In/out flow of x-momentum

$$\oint_S u\boldsymbol{u}\cdot\boldsymbol{n}dS$$



Integrate over the boundary

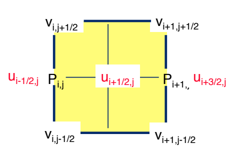
$$\oint_S u\boldsymbol{u}\cdot\boldsymbol{n}dS \approx$$

$$\left((u^2)^n_{i+1,j} - (u^2)^n_{i,j} + (uv)^n_{i+1/2,j+1/2} - (uv)^n_{i+1/2,j-1/2}\right)h$$

Computational Fluid Dynamics
Advection terms

Find the fluxes

Interpolate the velocities as needed



$$(u^2)^n_{i+1,j} = \left(\frac{1}{2}(u^n_{i+3/2,j} + u^n_{i+1/2,j})\right)^2$$

$$(u^2)^n_{i,j} = \left(\frac{1}{2}(u^n_{i+1/2,j} + u^n_{i-1/2,j})\right)^2$$

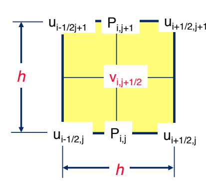
$$(uv)^n_{i+1/2,j+1/2} = \left(\frac{1}{2}(u^n_{i+1/2,j} + u^n_{i+1/2,j+1})\right)\left(\frac{1}{2}(v^n_{i,j+1/2} + v^n_{i+1,j+1/2})\right)$$

$$(uv)^n_{i+1/2,j-1/2} = \left(\frac{1}{2}(u^n_{i+1/2,j} + u^n_{i+1/2,j-1})\right)\left(\frac{1}{2}(v^n_{i,j-1/2} + v^n_{i+1,j-1/2})\right)$$

Computational Fluid Dynamics
Advection terms

In/out flow of y-momentum

$$\oint_S v\boldsymbol{u}\cdot\boldsymbol{n}dS$$



Integrate over the boundary

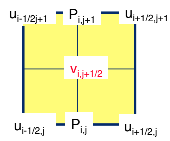
$$\oint_S v\boldsymbol{u}\cdot\boldsymbol{n}dS \approx$$

$$\left((uv)^n_{i+1/2,j+1/2} - (uv)^n_{i-1/2,j+1/2} + (v^2)^n_{i,j+1} - (v^2)^n_{i,j}\right)h$$

Computational Fluid Dynamics
Advection terms

Find the fluxes

Interpolate the velocities as needed



$$(v^2)^n_{i,j+1} = \left(\frac{1}{2}(v^n_{i,j+3/2} + v^n_{i,j+1/2})\right)^2$$

$$(v^2)^n_{i,j} = \left(\frac{1}{2}(v^n_{i,j+1/2} + v^n_{i,j-1/2})\right)^2$$

$$(uv)^n_{i+1/2,j+1/2} = \left(\frac{1}{2}(u^n_{i+1/2,j} + u^n_{i+1/2,j+1})\right)\left(\frac{1}{2}(v^n_{i,j+1/2} + v^n_{i+1,j+1/2})\right)$$

$$(uv)^n_{i-1/2,j+1/2} = \left(\frac{1}{2}(u^n_{i-1/2,j} + u^n_{i-1/2,j+1})\right)\left(\frac{1}{2}(v^n_{i,j+1/2} + v^n_{i-1,j+1/2})\right)$$

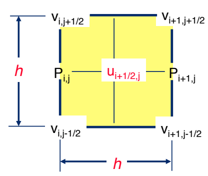
Computational Fluid Dynamics

Conservation of Momentum
The pressure term

Computational Fluid Dynamics
Pressure term

Pressure force in the x-direction

$$\tfrac{1}{\rho}\oint_S p n_x dS$$



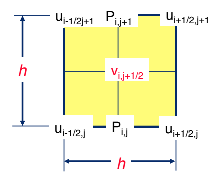
Integrate over the boundary

$$\tfrac{1}{\rho}\oint_S p n_x dS \approx \frac{1}{\rho}\left(p_{i+1,j} - p_{i,j}\right)h$$

Computational Fluid Dynamics
Pressure term

Pressure force in the y-direction

$$\tfrac{1}{\rho}\oint_S p n_y dS$$



Integrate over the boundary

$$\tfrac{1}{\rho}\oint_S p n_y dS \approx \frac{1}{\rho}\left(p_{i,j+1} - p_{i,j}\right)h$$

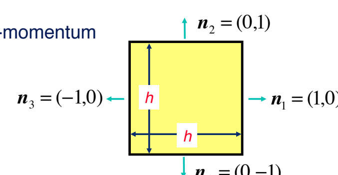
Computational Fluid Dynamics

Conservation of Momentum
The viscous terms

Computational Fluid Dynamics
Viscous terms—x-momentum

Viscous diffusion of x-momentum

Approximate the integral:

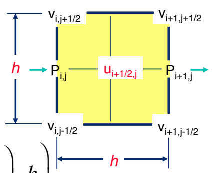


$$\nu\oint_S \nabla u \cdot \boldsymbol{n} dS = \nu\oint_S \left(\frac{\partial u}{\partial x}n_x + \frac{\partial u}{\partial y}n_y\right)dS$$

$$\approx \nu\left(\left(\frac{\partial u}{\partial x}\right)_1 h + \left(\frac{\partial u}{\partial y}\right)_2 h - \left(\frac{\partial u}{\partial x}\right)_3 h - \left(\frac{\partial u}{\partial y}\right)_4 h\right)$$

Computational Fluid Dynamics
Viscous terms—x-momentum

Viscous diffusion of x-momentum



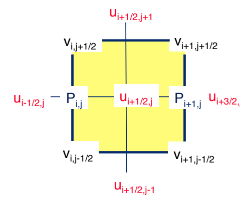
$$\nu\oint_S \nabla u \cdot \boldsymbol{n}\, dS$$

$$\approx \nu\left(\left(\frac{\partial u}{\partial x}\right)_1 h + \left(\frac{\partial u}{\partial y}\right)_2 h - \left(\frac{\partial u}{\partial x}\right)_3 h - \left(\frac{\partial u}{\partial y}\right)_4 h\right)$$

$$= \left(\left(\frac{\partial u}{\partial x}\right)^n_{i+1,j} - \left(\frac{\partial u}{\partial x}\right)^n_{i,j} + \left(\frac{\partial u}{\partial y}\right)^n_{i+1/2,j+1/2} - \left(\frac{\partial u}{\partial y}\right)^n_{i+1/2,j-1/2}\right)h$$

Computational Fluid Dynamics
Viscous terms—x-momentum

Computing the derivatives at the boundary:



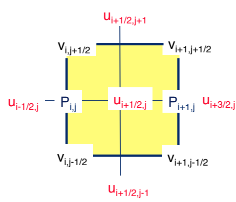
$$\left(\frac{\partial u}{\partial x}\right)^n_{i,j} \approx \frac{u^n_{i+1/2,j} - u^n_{i-1/2,j}}{h}$$

Substitute:

$$\left(\left(\frac{\partial u}{\partial x}\right)^n_{i+1,j} - \left(\frac{\partial u}{\partial x}\right)^n_{i,j}\right)h \approx \left(\frac{u^n_{i+3/2,j} + u^n_{i-1/2,j} - 2u^n_{i+1/2,j}}{\cancel{h}}\right)\cancel{h}$$

Computational Fluid Dynamics
Viscous terms—x-momentum

Computing the derivatives at the boundary:



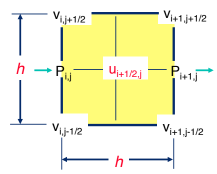
$$\left(\frac{\partial u}{\partial y}\right)^n_{i+1/2,j+1/2} \approx \frac{u^n_{i+1/2,j+1} - u^n_{i-1/2,j}}{h}$$

Substitute:

$$\left(\left(\frac{\partial u}{\partial y}\right)^n_{i+1/2,j+1/2} - \left(\frac{\partial u}{\partial y}\right)^n_{i+1/2,j-1/2}\right)h \approx \left(\frac{u^n_{i+1/2,j+1} + u^n_{i+1/2,j-1} - 2u^n_{i+1/2,j}}{\cancel{h}}\right)\cancel{h}$$

Computational Fluid Dynamics
Viscous terms—x-momentum

x-diffusion of momentum



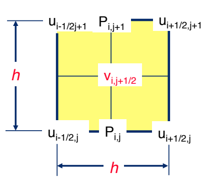
The final result is:

$$\nu\oint_S \nabla u \cdot \boldsymbol{n}\, dS \approx$$

$$\nu\left(u^n_{i+3/2,j} + u^n_{i-1/2,j} + u^n_{i+1/2,j+1} + u^n_{i+1/2,j-1} - 4u^n_{i+1/2,j}\right)$$

Computational Fluid Dynamics
Viscous terms—y-momentum

Similarly:
y-diffusion of momentum



The final result is:

$$\nu\oint_S \nabla v \cdot \boldsymbol{n}\, dS \approx$$

$$\nu\left(v^n_{i,j+3/2} + v^n_{i,j-1/2} + v^n_{i+1,j+1/2} + v^n_{i-1,j+1/2} - 4v^n_{i,j+1/2}\right)$$

Computational Fluid Dynamics

Putting it together

Computational Fluid Dynamics
Gathering the terms

$$\frac{\partial}{\partial t}\int_V u\,dV = -\oint_S u\mathbf{u}\cdot\mathbf{n}dS + \nu\oint_S \nabla u\cdot\mathbf{n}dS - \tfrac{1}{\rho}\oint_S p n_x dS$$

$$\frac{u^{n+1}_{i+1/2,j}-u^n_{i+1/2,j}}{\Delta t} = \frac{-1}{h}\left((u^2)^n_{i+1,j}-(u^2)^n_{i,j}+(uv)^n_{i+1/2,j+1/2}-(uv)^n_{i+1/2,j-1/2}\right)$$
$$+\frac{\nu}{h^2}\left(u^n_{i+3/2,j}+u^n_{i-1/2,j}+u^n_{i+1/2,j+1}+u^n_{i+1/2,j-1}-4u^n_{i+1/2,j}\right)$$
$$-\frac{1}{h}\left(P_{i+1,j}-P_{i,j}\right)$$

Where $P=\dfrac{p}{\rho}$

Computational Fluid Dynamics
Gathering the terms

$$\frac{\partial}{\partial t}\int_V v\,dV = -\oint_S v\mathbf{u}\cdot\mathbf{n}dS + \nu\oint_S \nabla v\cdot\mathbf{n}dS - \tfrac{1}{\rho}\oint_S p n_y dS$$

$$\frac{v^{n+1}_{i,j+1/2}-v^n_{i,j+1/2}}{\Delta t} = \frac{-1}{h}\left((uv)^n_{i+1/2,j+1/2}-(uv)^n_{i-1/2,j+1/2}+(v^2)^n_{i,j+1}-(v^2)^n_{i,j}\right)$$
$$+\frac{\nu}{h^2}\left(v^n_{i+1,j+1/2}+v^n_{i-1,j+1/2}+v^n_{i,j+3/2}+v^n_{i+1/2,j-1/2}-4v^n_{i,j+1/2}\right)$$
$$-\frac{1}{h}\left(P_{i,j+1}-P_{i,j}\right)$$

Where $P=\dfrac{p}{\rho}$

Computational Fluid Dynamics
Summary

$$\frac{u^{n+1}_{i+1/2,j}-u^n_{i+1/2,j}}{\Delta t} = \frac{-1}{h}\left((u^2)^n_{i+1,j}-(u^2)^n_{i,j}+(uv)^n_{i+1/2,j+1/2}-(uv)^n_{i+1/2,j-1/2}\right)$$
$$+\frac{\nu}{h^2}\left(u^n_{i+3/2,j}+u^n_{i-1/2,j}+u^n_{i+1/2,j+1}+u^n_{i+1/2,j-1}-4u^n_{i+1/2,j}\right)$$
$$-\frac{1}{h}\left(P_{i+1,j}-P_{i,j}\right)$$

$$\frac{v^{n+1}_{i,j+1/2}-v^n_{i,j+1/2}}{\Delta t} = \frac{-1}{h}\left((uv)^n_{i+1/2,j+1/2}-(uv)^n_{i-1/2,j+1/2}+(v^2)^n_{i,j+1}-(v^2)^n_{i,j}\right)$$
$$+\frac{\nu}{h^2}\left(v^n_{i+1,j+1/2}+v^n_{i-1,j+1/2}+v^n_{i,j+3/2}+v^n_{i+1/2,j-1/2}-4v^n_{i,j+1/2}\right)$$
$$-\frac{1}{h}\left(P_{i,j+1}-P_{i,j}\right)$$

$$u^{n+1}_{i+1/2,j}-u^{n+1}_{i-1/2,j}+v^{n+1}_{i,j+1/2}-v^{n+1}_{i,j-1/2}=0$$

Computational Fluid Dynamics

Solution Strategy

Computational Fluid Dynamics

Momentum equations

$$\frac{u^{n+1}_{i+1/2,j}-u^n_{i+1/2,j}}{\Delta t} = \frac{-1}{h}\left((u^2)^n_{i+1,j}-(u^2)^n_{i,j}+(uv)^n_{i+1/2,j+1/2}-(uv)^n_{i+1/2,j-1/2}\right)$$
$$-\frac{1}{h}\left(P_{i+1,j}-P_{i,j}\right)+\frac{\nu}{h^2}\left(u^n_{i+3/2,j}+u^n_{i-1/2,j}+u^n_{i+1/2,j+1}+u^n_{i+1/2,j-1}-4u^n_{i+1/2,j}\right)$$

$$\frac{v^{n+1}_{i,j+1/2}-v^n_{i,j+1/2}}{\Delta t} = \frac{-1}{h}\left((uv)^n_{i+1/2,j+1/2}-(uv)^n_{i-1/2,j+1/2}+(v^2)^n_{i,j+1}-(v^2)^n_{i,j}\right)$$
$$-\frac{1}{h}\left(P_{i,j+1}-P_{i,j}\right)+\frac{\nu}{h^2}\left(v^n_{i+1,j+1/2}+v^n_{i-1,j+1/2}+v^n_{i,j+3/2}+v^n_{i+1/2,j-1/2}-4v^n_{i,j+1/2}\right)$$

Write as one vector equation

$$\frac{\mathbf{u}^{n+1}_{i,j}-\mathbf{u}^n_{i,j}}{\Delta t} = -\mathbf{A}^n_{i,j}-\nabla_h P_{i,j}+\mathbf{D}^n_{i,j}$$

Computational Fluid Dynamics

Mass conservation equation

$$u^{n+1}_{i+1/2,j}-u^{n+1}_{i-1/2,j}+v^{n+1}_{i,j+1/2}-v^{n+1}_{i,j-1/2}=0$$

Write as one vector equation

$$\nabla_h\cdot\mathbf{u}^{n+1}_{i,j}=0$$

Where we have used

$$\nabla\cdot\boldsymbol{u} = \lim_{\delta V\to 0}\frac{1}{\delta V}\oint_S \boldsymbol{u}\cdot\boldsymbol{n}\,ds$$

to define the numerical divergence

Computational Fluid Dynamics
Discretization in time

Summary of discrete vector equations

$$\boxed{\frac{\mathbf{u}^{n+1}_{i,j}-\mathbf{u}^{n}_{i,j}}{\Delta t} = -\mathbf{A}^{n}_{i,j} - \nabla_h P_{i,j} + \mathbf{D}^{n}_{i,j}}$$ Evolution of the velocity

$$\boxed{\nabla_h \cdot \mathbf{u}^{n+1}_{i,j} = 0}$$ Constraint on velocity

No explicit equation for the pressure!

Computational Fluid Dynamics
Discretization in time

Split

$$\frac{\mathbf{u}^{n+1}_{i,j}-\mathbf{u}^{n}_{i,j}}{\Delta t} = -\mathbf{A}^{n}_{i,j} - \nabla_h P_{i,j} + \mathbf{D}^{n}_{i,j}$$

into

$$\frac{\mathbf{u}^{t}_{i,j}-\mathbf{u}^{n}_{i,j}}{\Delta t} = -\mathbf{A}^{n}_{i,j} + \mathbf{D}^{n}_{i,j} \quad \Rightarrow \quad \mathbf{u}^{t}_{i,j} = \mathbf{u}^{n}_{i,j} + \Delta t\left(-\mathbf{A}^{n}_{i,j} + \mathbf{D}^{n}_{i,j}\right)$$

and

$$\frac{\mathbf{u}^{n+1}_{i,j}-\mathbf{u}^{t}_{i,j}}{\Delta t} = -\nabla_h P_{i,j} \quad \Rightarrow \quad \mathbf{u}^{n+1}_{i,j} = \mathbf{u}^{t}_{i,j} - \Delta t\, \nabla_h P_{i,j}$$

by introducing the temporary velocity $\mathbf{u}^{t}$

Projection Method

Computational Fluid Dynamics
Discretization in time

To derive an equation for the pressure we take the divergence of

$$\mathbf{u}^{n+1}_{i,j} = \mathbf{u}^{t}_{i,j} - \Delta t\, \nabla_h P_{i,j}$$

and use the mass conservation equation

$$\nabla_h \cdot \mathbf{u}^{n+1}_{i,j} = 0$$

The result is

$$\nabla_h \cdot \cancelto{0}{\mathbf{u}^{n+1}_{i,j}} = \nabla_h \cdot \mathbf{u}^{t}_{i,j} - \Delta t\, \nabla_h \cdot \nabla_h P_{i,j}$$

$$\longrightarrow \nabla^2_h P_{i,j} = \frac{1}{\Delta t} \nabla_h \cdot \mathbf{u}^{t}_{i,j}$$

Computational Fluid Dynamics
Discretization in time

1. Find a temporary velocity using the advection and the diffusion terms only:

$$\boxed{\mathbf{u}^{t}_{i,j} = \mathbf{u}^{n}_{i,j} + \Delta t\left(-\mathbf{A}^{n}_{i,j} + \mathbf{D}^{n}_{i,j}\right)}$$

2. Find the pressure needed to make the velocity field incompressible

$$\boxed{\nabla^2_h P_{i,j} = \frac{1}{\Delta t} \nabla_h \cdot \mathbf{u}^{t}_{i,j}}$$

3. Correct the velocity by adding the pressure gradient:

$$\boxed{\mathbf{u}^{n+1}_{i,j} = \mathbf{u}^{t}_{i,j} - \Delta t\, \nabla_h P_{i,j}}$$

Computational Fluid Dynamics
Algorithm

Initial field given

Determine u, v boundary conditions

Advect $\mathbf{u}^{t}_{i,j} = \mathbf{u}^{n}_{i,j} + \Delta t\left(-\mathbf{A}^{n}_{i,j} + \mathbf{D}^{n}_{i,j}\right)$

Poisson equation for $P_{i,j}$ (SOR)

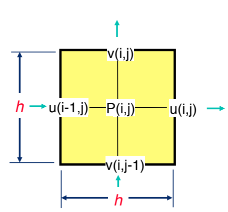
Projection $\mathbf{u}^{n+1}_{i,j} = \mathbf{u}^{t}_{i,j} - \Delta t\, \nabla_h P_{i,j}$

t=t+Δt

Computational Fluid Dynamics
Computational Grid

Since a fractional number is not allowed in computer program, redefine velocity node indices:

$$u(i,j) = u_{i+1/2,j}$$
$$v(i,j) = v_{i,j+1/2}$$



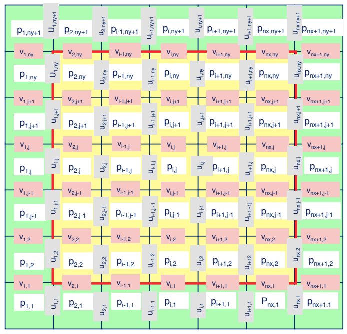
Write also

$$u(i,j) = u_{i,j}$$
$$v(i,j) = v_{i,j}$$

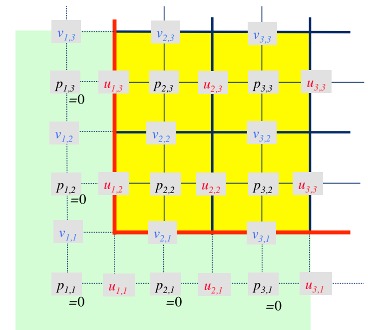
Computational Fluid Dynamics
Computational Grid

Array Dimension:

$P(1:nx+1, 1:ny+1)$
$u(1:nx, 1:ny+1)$
$v(1:nx+1, 1:ny)$



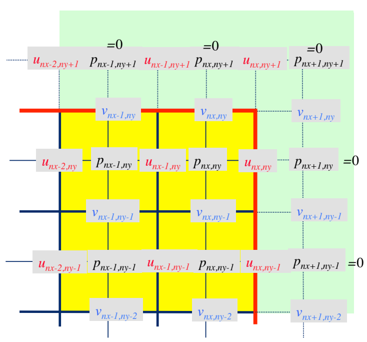
Computational Fluid Dynamics
Boundary Grid Structure



Array Dimension:

$P(1:nx+1, 1:ny+1)$
$u(1:nx, 1:ny+1)$
$v(1:nx+1, 1:ny)$

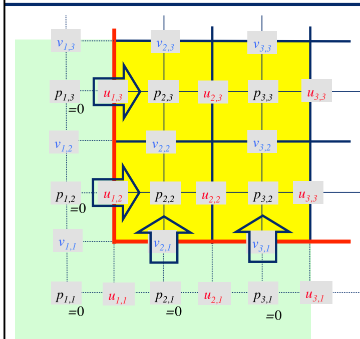
Computational Fluid Dynamics
Boundary Grid Structure



Array Dimension:

$P(1:nx+1, 1:ny+1)$
$u(1:nx, 1:ny+1)$
$v(1:nx+1, 1:ny)$

Computational Fluid Dynamics



The edges of the pressure control volumes coincide with the boundaries of the computational domain. The normal velocities are therefore specified directly on the walls

Computational Fluid Dynamics

Boundary conditions for the tangential velocity

Computational Fluid Dynamics
Velocity Boundary Conditions

Velocity of wall is given, U_{wall} (no-slip)

Interpolate linearly

$$\frac{u_{i,2}+u_{i,1}}{2}=U_{wall}$$

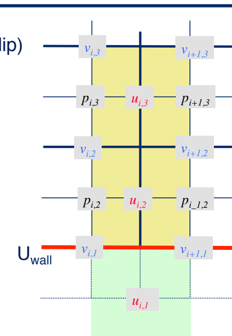
Solve for the “ghost” velocity

$$u_{i,1}=2U_{wall}-u_{i,2}$$

If $U_{wall}=0$

$$u_{i,1}=-u_{i,2}$$

(reflection technique)



Computational Fluid Dynamics

The Pressure Equation

Computational Fluid Dynamics

$$u^{n+1}_{i+1/2,j} = u^{t}_{i+1/2,j} - \frac{\Delta t}{h}\left(P_{i+1,j} - P_{i,j}\right)$$

$$u^{n+1}_{i-1/2,j} = u^{t}_{i-1/2,j} - \frac{\Delta t}{h}\left(P_{i,j} - P_{i-1,j}\right)$$

$$v^{n+1}_{i,j+1/2} = v^{t}_{i,j+1/2} - \frac{\Delta t}{h}\left(P_{i,j+1} - P_{i,j}\right)$$

$$v^{n+1}_{i,j-1/2} = v^{t}_{i,j-1/2} - \frac{\Delta t}{h}\left(P_{i,j} - P_{i,j-1}\right)$$

Substitute into:

$$u^{n+1}_{i+1/2,j} - u^{n+1}_{i-1/2,j} + v^{n+1}_{i,j+1/2} - v^{n+1}_{i,j-1/2} = 0$$

Giving:

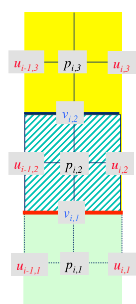
$$P_{i+1,j} + P_{i-1,j} + P_{i,j+1} + P_{i,j-1} - 4P_{i,j} = \frac{h}{\Delta t}\left(u^{t}_{i+1/2,j} - u^{t}_{i-1/2,j} + v^{t}_{i,j+1/2} - v^{t}_{i,j-1/2}\right)$$

Computational Fluid Dynamics
Boundary Conditions - Pressure

Apply continuity at the boundary:

$$u^{n+1}_{i,2} - u^{n+1}_{i-1,2} + v^{n+1}_{i,2} - \cancelto{0}{v^{n+1}_{i,1}} = 0$$

$$\rightarrow \quad u^{n+1}_{i,2} - u^{n+1}_{i-1,2} + v^{n+1}_{i,2} = 0$$



Substitute for the velocities

$$v^{n+1}_{i,2} = v^{t}_{i,2} - \frac{\Delta t}{h}\left(P_{i,3} - P_{i,2}\right)$$

$$u^{n+1}_{i,2} = u^{t}_{i,2} - \frac{\Delta t}{h}\left(P_{i+1,2} - P_{i,2}\right)$$

$$u^{n+1}_{i-1,2} = u^{t}_{i-1,2} - \frac{\Delta t}{h}\left(P_{i,2} - P_{i-1,2}\right)$$

Computational Fluid Dynamics
Boundary Conditions - Pressure

$$v^{n+1}_{i,2} = v^{t}_{i,2} - \frac{\Delta t}{h}\left(P_{i,3} - P_{i,2}\right)$$

$$u^{n+1}_{i,2} = u^{t}_{i,2} - \frac{\Delta t}{h}\left(P_{i+1,2} - P_{i,2}\right)$$

$$u^{n+1}_{i-1,2} = u^{t}_{i-1,2} - \frac{\Delta t}{h}\left(P_{i,2} - P_{i-1,2}\right)$$

Substitute into:

$$u^{n+1}_{i,2} - u^{n+1}_{i-1,2} + v^{n+1}_{i,2} = 0$$

Giving:

$$u^{t}_{i,2} - \frac{\Delta t}{h}\left(P_{i+1,2} - P_{i,2}\right) - u^{t}_{i-1,2} + \frac{\Delta t}{h}\left(P_{i,2} - P_{i-1,2}\right) + v^{t}_{i,2} - \frac{\Delta t}{h}\left(P_{i,3} - P_{i,2}\right) = 0$$

Rearrange

$$P_{i+1,2} + P_{i-1,2} + P_{i,3} - 3P_{i,2} = \frac{h}{\Delta t}\left(u^{t}_{i,2} - u^{t}_{i-1,2} + v^{t}_{i,2}\right)$$

Computational Fluid Dynamics
Solving the pressure equation

Solving for the pressure $\quad P(i,j) \quad i = 2,nx; \; j = 2,ny$

Interior nodes: $\quad i = 3,\cdots,nx-1; \; j = 3,\cdots,ny-1$

$$P_{i,j} = \frac{1}{4}\beta\left(\left(P_{i+1,j} + P_{i-1,j} + P_{i,j+1} + P_{i,j-1}\right) - \frac{h}{\Delta t}\left(u^{t}_{i,j} - u^{t}_{i-1,j} + v^{t}_{i,j} - v^{t}_{i,j-1}\right)\right) + (1-\beta)P_{i,j}$$

Boundary nodes except corner: $\quad i = 2; i = nx; \; j = 2; \; j = ny$

$$P_{i,j} = \frac{1}{3}\beta\left(\underbrace{\left(P_{i+1,j} + P_{i-1,j} + P_{i,j+1} + P_{i,j-1}\right)} - \frac{h}{\Delta t}\underbrace{\left(u^{t}_{i,j} - u^{t}_{i-1,j} + v^{t}_{i,j} - v^{t}_{i,j-1}\right)}\right) + (1-\beta)P_{i,j}$$

One is zero

Corner nodes: $(i,j) = (2,2); (nx,2); (2,ny); (nx,ny)$

$$P_{i,j} = \frac{1}{2}\beta\left(\underbrace{\left(P_{i+1,j} + P_{i-1,j} + P_{i,j+1} + P_{i,j-1}\right)} - \frac{h}{\Delta t}\underbrace{\left(u^{t}_{i,j} - u^{t}_{i-1,j} + v^{t}_{i,j} - v^{t}_{i,j-1}\right)}\right) + (1-\beta)P_{i,j}$$

Two are zero

Computational Fluid Dynamics

The Structure of the Code

Computational Fluid Dynamics
Algorithm

Initial field given

Determine u, v boundary conditions

$$\mathbf{u}^{t}_{i,j} = \mathbf{u}^{n}_{i,j} + \Delta t\left(-\mathbf{A}^{n}_{i,j} + \mathbf{D}^{n}_{i,j}\right)$$

Poisson equation for $P_{i,j}$ (SOR)

$$\mathbf{u}^{n+1}_{i,j} = \mathbf{u}^{t}_{i,j} - \Delta t \, \nabla_h P_{i,j}$$

t=t+Δt

```
Initialize parameters and arrays, set
time step

for is=1:nstep
   set boundary conditions for
   tangential velocity (ghost
   points)

   Find predicted velocity

   Solve for pressure using SOR

   Find the projected velocity
   by adding the pressure gradient

   PLOT
end
```

Computational Fluid Dynamics

Discrete predicted velocities

```
ut(i,j)=u(i,j)+dt*(-(0.25/h)*((u(i+1,j)+u(i,j))^2-(u(i,j)+...
       u(i-1,j))^2+(u(i,j+1)+u(i,j))*(v(i+1,j)+...
       v(i,j))-(u(i,j)+u(i,j-1))*(v(i+1,j-1)+v(i,j-1)))+...
      (mu/h^2)*(u(i+1,j)+u(i-1,j)+u(i,j+1)+u(i,j-1)-4*u(i,j)));

vt(i,j)=v(i,j)+dt*(-(0.25/h)*((u(i,j+1)+u(i,j))*(v(i+1,j)+...
       v(i,j))-(u(i-1,j+1)+u(i-1,j))*(v(i,j)+v(i-1,j))+...
       (v(i,j+1)+v(i,j))^2-(v(i,j)+v(i,j-1))^2)+...
       (mu/h^2)*(v(i+1,j)+v(i-1,j)+v(i,j+1)+v(i,j-1)-4*v(i,j)));
```

Discrete corrected velocities

```
u(2:nx,2:ny+1)=...
         ut(2:nx,2:ny+1)-(dt/h)*(p(3:nx+1,2:ny+1)-p(2:nx,2:ny+1));
v(2:nx+1,2:ny)=...
         vt(2:nx+1,2:ny)-(dt/h)*(p(2:nx+1,3:ny+1)-p(2:nx+1,2:ny));
```

Computational Fluid Dynamics
Solving the pressure equation

$$P_{i,j} = \frac{1}{4}\beta\left(\left(P_{i+1,j} + P_{i-1,j} + P_{i,j+1} + P_{i,j-1}\right) - \frac{h}{\Delta t}\left(u^{t}_{i,j} - u^{t}_{i-1,j} + v^{t}_{i,j} - v^{t}_{i,j-1}\right)\right) + (1-\beta)P_{i,j}$$

$$P_{i,j} = \frac{1}{3}\beta\left(\left(P_{i+1,j} + P_{i-1,j} + P_{i,j+1} + P_{i,j-1}\right) - \frac{h}{\Delta t}\left(u^{t}_{i,j} - u^{t}_{i-1,j} + v^{t}_{i,j} - v^{t}_{i,j-1}\right)\right) + (1-\beta)P_{i,j}$$

$$P_{i,j} = \frac{1}{2}\beta\left(\left(P_{i+1,j} + P_{i-1,j} + P_{i,j+1} + P_{i,j-1}\right) - \frac{h}{\Delta t}\left(u^{t}_{i,j} - u^{t}_{i-1,j} + v^{t}_{i,j} - v^{t}_{i,j-1}\right)\right) + (1-\beta)P_{i,j}$$

Pressure

```
p(i,j)=beta*c(i,j)*(p(i+1,j)+p(i-1,j)+p(i,j+1)+p(i,j-1)-...
       (h/dt)*(ut(i,j)-ut(i-1,j)+vt(i,j)-vt(i,j-1)))+(1-beta)*p(i,j);
```

where

c(i,j)=1/4 for interior nodes; c(i,j)=1/3 for boundary nodes;
c(i,j)=1/2 for corner nodes;
Set p(i,j)=0 for ghost points.
Set the intermediate normal velocities at the wall to zero

Computational Fluid Dynamics

THE CODE
Matlab Implementation

Computational Fluid Dynamics

```
nx=9;ny=9;dt=0.02;nstep=200;mu=0.1;maxit=100;beta=1.2;h=1/nx;
u=zeros(nx+1,ny+2);v=zeros(nx+2,ny+1);p=zeros(nx+2,ny+2);
ut=zeros(nx+1,ny+2);vt=zeros(nx+2,ny+1);c=zeros(nx+2,ny+2)+0.25;
uu=zeros(nx+1,ny+1);vv=zeros(nx+1,ny+1);w=zeros(nx+1,ny+1);
c(2,3:ny)=1/3;c(nx+1,3:ny)=1/3;c(3:nx,2)=1/3;c(3:nx,ny+1)=1/3;
c(2,2)=1/2;c(2,ny+1)=1/2;c(nx+1,2)=1/2;c(nx+1,ny+1)=1/2;
un=1;us=0;ve=0;vw=0;time=0.0;

for is=1:nstep
   u(1:nx+1,1)=2*us-u(1:nx+1,2);u(1:nx+1,ny+2)=2*un-u(1:nx+1,ny+1);
   v(1,1:ny+1)=2*vw-v(2,1:ny+1);v(nx+2,1:ny+1)=2*ve-v(nx+1,1:ny+1)
```

Computational Fluid Dynamics

```
for i=2:nx, for j=2:ny+1          % temporary u-velocity
  ut(i,j)=u(i,j)+dt*(-(0.25/h)*((u(i+1,j)+u(i,j))^2-(u(i,j)+...
     u(i-1,j))^2+(u(i,j+1)+u(i,j))*(v(i+1,j)+...
     v(i,j))-(u(i,j)+u(i,j-1))*(v(i+1,j-1)+v(i,j-1)))+...
    (mu/h^2)*(u(i+1,j)+u(i-1,j)+u(i,j+1)+u(i,j-1)-4*u(i,j)));
end, end

for i=2:nx+1, for j=2:ny          % temporary v-velocity
  vt(i,j)=v(i,j)+dt*(-(0.25/h)*((u(i,j+1)+u(i,j))*(v(i+1,j)+...
     v(i,j))-(u(i-1,j+1)+u(i-1,j))*(v(i,j)+v(i-1,j))+...
     (v(i,j+1)+v(i,j))^2-(v(i,j)+v(i,j-1))^2)+...
     (mu/h^2)*(v(i+1,j)+v(i-1,j)+v(i,j+1)+v(i,j-1)-4*v(i,j)));
end, end
```

Computational Fluid Dynamics

```
for it=1:maxit                    % solve for pressure
  for i=2:nx+1, for j=2:ny+1
     p(i,j)=beta*c(i,j)*(p(i+1,j)+p(i-1,j)+p(i,j+1)+p(i,j-1)-...
     (h/dt)*(ut(i,j)-ut(i-1,j)+vt(i,j)-vt(i,j-1)))+(1-beta)*p(i,j);
  end, end
end
                             % correct the velocity
u(2:nx,2:ny+1)=...
         ut(2:nx,2:ny+1)-(dt/h)*(p(3:nx+1,2:ny+1)-p(2:nx,2:ny+1));
v(2:nx+1,2:ny)=...
         vt(2:nx+1,2:ny)-(dt/h)*(p(2:nx+1,3:ny+1)-p(2:nx+1,2:ny));

time=time+dt
```

Computational Fluid Dynamics

```
% plot results
   uu(1:nx+1,1:ny+1)=0.5*(u(1:nx+1,2:ny+2)+u(1:nx+1,1:ny+1));
   vv(1:nx+1,1:ny+1)=0.5*(v(2:nx+2,1:ny+1)+v(1:nx+1,1:ny+1));
   w(1:nx+1,1:ny+1)=(u(1:nx+1,2:ny+2)-u(1:nx+1,1:ny+1)-...
           v(2:nx+2,1:ny+1)+v(1:nx+1,1:ny+1))/(2*h);
   hold off,quiver(uu',vv','r');
   hold on;contour(w',20),axis equal,pause(0.01)
end
```

Limitations on the time step

$$\frac{\nu \Delta t}{h^2} \le \frac{1}{4} \qquad \frac{(|u|+|v|)\Delta t}{\nu} \le 2$$

Computational Fluid Dynamics

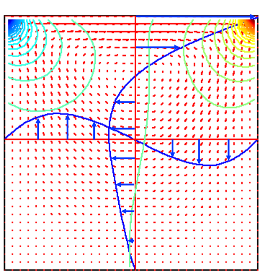
```
nx=16;ny=16;dt=0.005;nstep=200;mu=0.1;maxit=100;beta=1.2;h=1/nx;
u=zeros(nx+1,ny+2);v=zeros(nx+2,ny+1);p=zeros(nx+2,ny+2);
ut=zeros(nx+1,ny+2);vt=zeros(nx+2,ny+1);c=zeros(nx+2,ny+2)+0.25;
uu=zeros(nx+1,ny+1);vv=zeros(nx+1,ny+1);w=zeros(nx+1,ny+1);
c(2,3:ny)=1/3;c(nx+1,3:ny)=1/3;c(3:nx,2)=1/3;c(3:nx,ny+1)=1/3;
c(2,2)=1/2;c(2,ny+1)=1/2;c(nx+1,2)=1/2;c(nx+1,ny+1)=1/2;
un=1;us=0;ve=0;vw=0;time=0.0;

for is=1:nstep
   u(1:nx+1,1)=2*us-u(1:nx+1,2);u(1:nx+1,ny+2)=2*un-u(1:nx+1,ny+1);
   v(1,1:ny+1)=2*vw-v(2,1:ny+1);v(nx+2,1:ny+1)=2*ve-v(nx+1,1:ny+1);

   for i=2:nx,for j=2:ny+1      % temporary u-velocity
     ut(i,j)=u(i,j)+dt*(-(0.25/h)*((u(i+1,j)+u(i,j))^2-(u(i,j)+...
         u(i-1,j))^2+(u(i,j+1)+u(i,j))*(v(i+1,j)+...
         v(i,j))-(u(i,j)+u(i,j-1))*(v(i+1,j-1)+v(i,j-1)))+...
        (mu/h^2)*(u(i+1,j)+u(i-1,j)+u(i,j+1)+u(i,j-1)-4*u(i,j)));
   end,end

   for i=2:nx+1,for j=2:ny      % temporary v-velocity
     vt(i,j)=v(i,j)+dt*(-(0.25/h)*((u(i,j+1)+u(i,j))*(v(i+1,j)+...
         v(i,j))-(u(i-1,j+1)+u(i-1,j))*(v(i,j)+v(i-1,j))+...
         (v(i,j+1)+v(i,j))^2-(v(i,j)+v(i,j-1))^2)+...
         (mu/h^2)*(v(i+1,j)+v(i-1,j)+v(i,j+1)+v(i,j-1)-4*v(i,j)));
   end,end

   for it=1:maxit                % solve for pressure
     for i=2:nx+1,for j=2:ny+1
        p(i,j)=beta*c(i,j)*(p(i+1,j)+p(i-1,j)+p(i,j+1)+p(i,j-1)-...
        (h/dt)*(ut(i,j)-ut(i-1,j)+vt(i,j)-vt(i,j-1)))+(1-beta)*p(i,j);
     end,end
   end
                                 % correct the velocity
   u(2:nx,2:ny+1)=ut(2:nx,2:ny+1)-(dt/h)*(p(3:nx+1,2:ny+1)-p(2:nx,2:ny+1));
   v(2:nx+1,2:ny)=vt(2:nx+1,2:ny)-(dt/h)*(p(2:nx+1,3:ny+1)-p(2:nx+1,2:ny));

   time=time+dt                  % plot the results
   uu(1:nx+1,1:ny+1)=0.5*(u(1:nx+1,2:ny+2)+u(1:nx+1,1:ny+1));
   vv(1:nx+1,1:ny+1)=0.5*(v(2:nx+2,1:ny+1)+v(1:nx+1,1:ny+1));
   w(1:nx+1,1:ny+1)=(u(1:nx+1,2:ny+2)-u(1:nx+1,1:ny+1)-...
           v(2:nx+2,1:ny+1)+v(1:nx+1,1:ny+1))/(2*h);
   hold off,quiver(uu',vv','r');
   hold on;contour(w',20),axis equal,pause(0.01)
end
```

Computational Fluid Dynamics

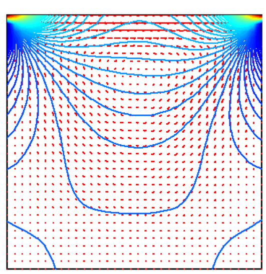
Results for the driven cavity

Computational Fluid Dynamics

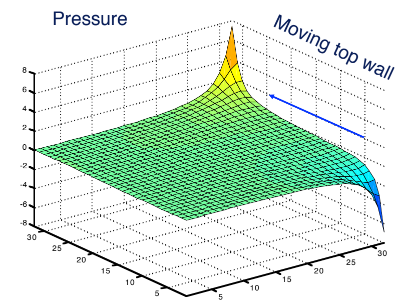
Velocity and pressure



Velocity and vorticity

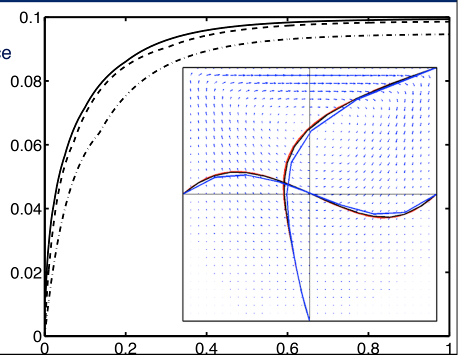


Computational Fluid Dynamics



Computational Fluid Dynamics

Convergence under grid refinement
8 × 8
16 × 16
32 × 32



Computational Fluid Dynamics

This code works. What more do we need?

For incompressible flows we need
- The ability to do complex 3D geometries
- Better advection for high Reynolds numbers
- Implicit viscous terms for low Reynolds numbers
- Fast pressure solvers

In addition, we need to deal with
- Compressible flows
- More complex physics

Computational Fluid Dynamics

More Complex Physics: Adding Temperature

Computational Fluid Dynamics

For incompressible flow the temperature equation can often be taken to be

$$\frac{\partial \rho c_p T}{\partial t} + \nabla \cdot \mathbf{u} \rho c_p T = \nabla \cdot k \nabla T$$

Which can be further simplified for constant material properties

$$\frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T = \alpha \nabla^2 T$$

The boundary conditions are usually that the temperature T_b or the normal gradient $\frac{\partial T_b}{\partial n}$ are given

The temperature is usually stored at the pressure nodes. Given wall temperature is implemented using ghost points.

Computational Fluid Dynamics

Diagnostics

Computational Fluid Dynamics

While looking at the solution is an important step in both understanding the flow and assessing the correctness of the solution, often we need more quantitative results, such as the total pressure and viscous force on a surface or an object.

The total force on a body is

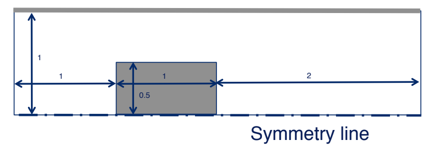
$$F_{Tot} = -\oint p\boldsymbol{n}\,ds + \oint 2\mu \mathbf{D} \cdot \boldsymbol{n}\,ds$$

The total heat transfer is

$$Q_{Tot} = \oint k \nabla T \cdot \boldsymbol{n}\,ds$$

Project II

Find the steady state force on a symmetric object by computing the flow around it.



Assume uniform inflow and outflow, with U=1. Take viscosity so that Re=20, based on the height of the channel and inlet velocity.

Report Format:

Title
Author, with affiliation
Introduction: Find 2-3 relevant references
Problem setup and numerical method: Discuss the new code structure and your solution strategy
Results: Present your plots
Discussion: Can be combined with Results or Conclusion
Conclusion: Sum up what done and what you learned

You can refer to the problem statement and do not to repeat it in your report. Your report should be as short as possible, but not shorter.