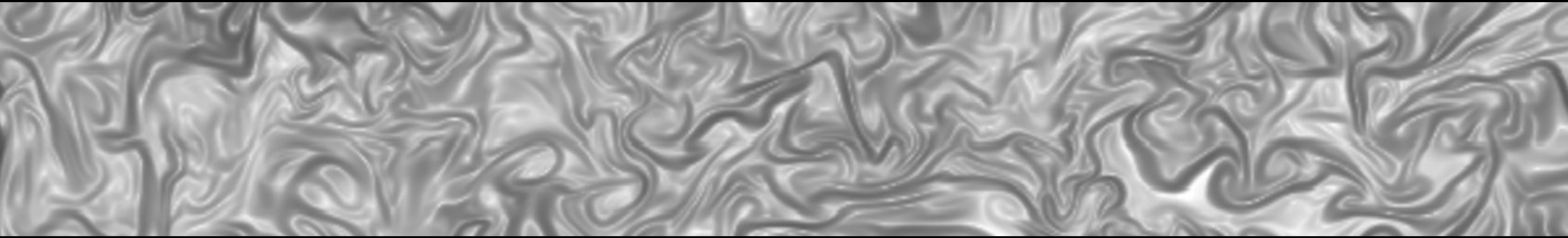
Incompressible NS solver with staggered grid

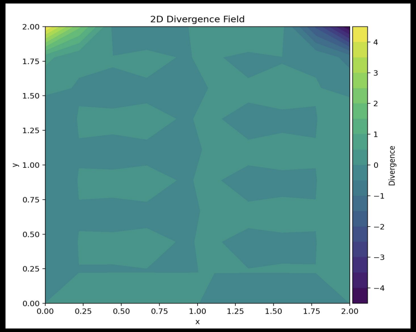
Update: 29th October 24’



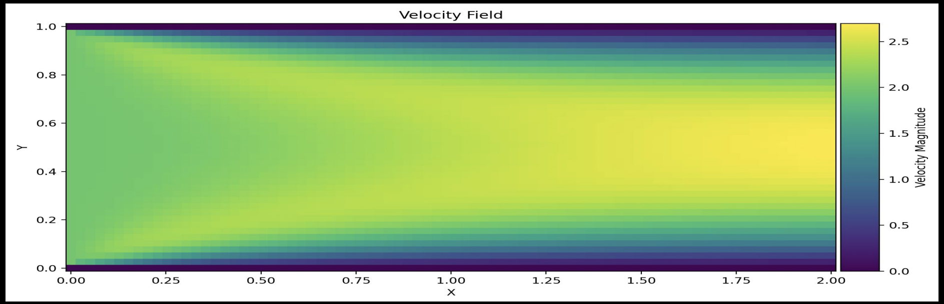
Issues with the collocated solver

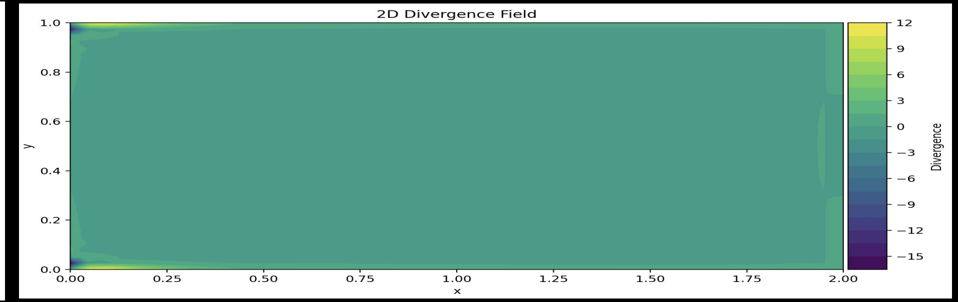
I can’t get rid of divergence…

Lid cavity

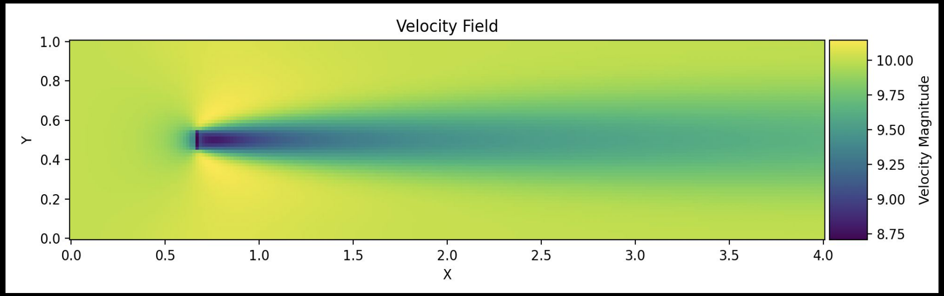


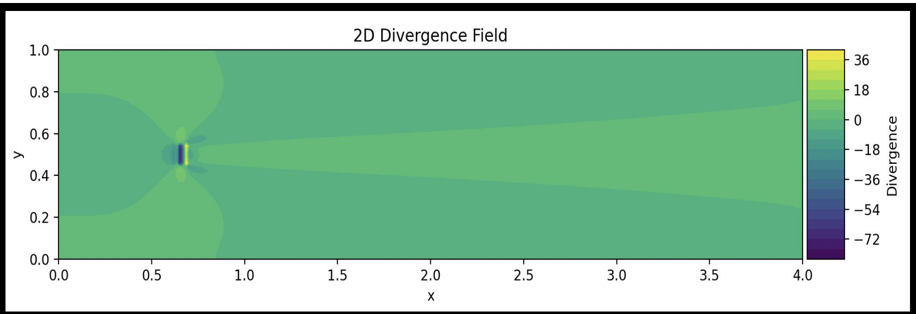
Channel flow





Wind turbine





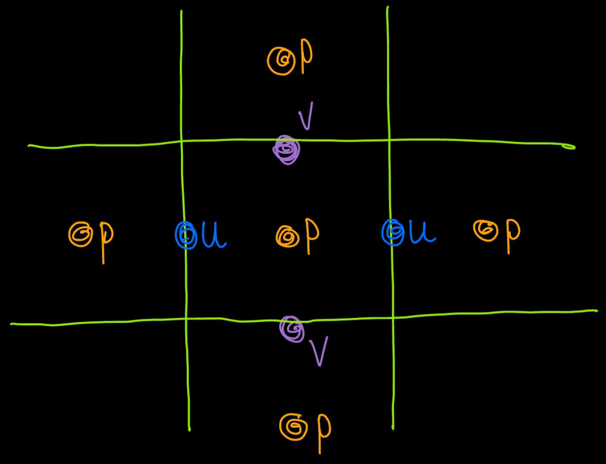
Staggered grid: a potential solution.

Pressure Gradient. The pressure gradient can be directly calculated between adjacent pressure points (which are at the cell centers). This avoids interpolation errors that occur in a collocated grid when computing the pressure gradient at the cell faces.

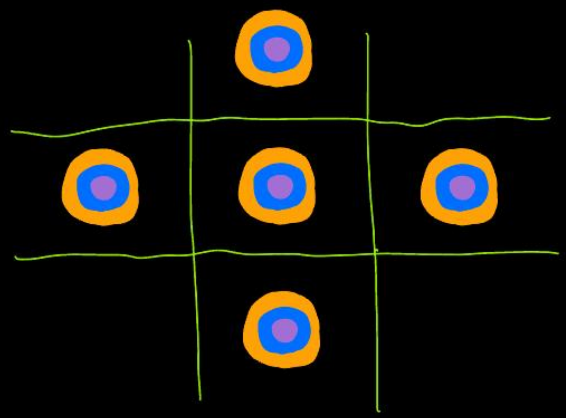
Divergence. The discretized divergence can be directly computed as the difference of face-centered velocities without the need for interpolation. This gives a more accurate representation of the fluxes in and out of the cell.

Pressure Poisson equation. Velocities are naturally aligned with the pressure cells in a way that ensures a better coupling between pressure and velocity during the solution of the pressure-Poisson equation.

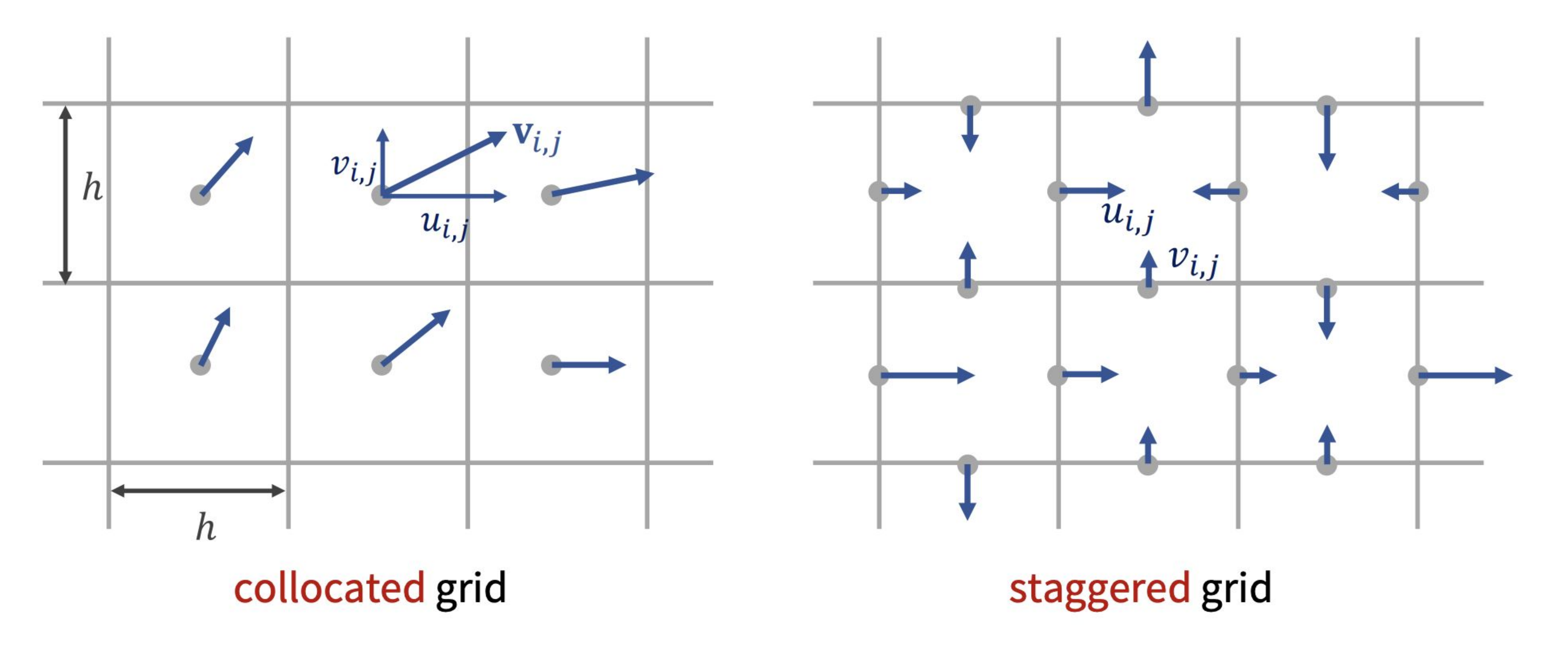
Staggered



Collocated



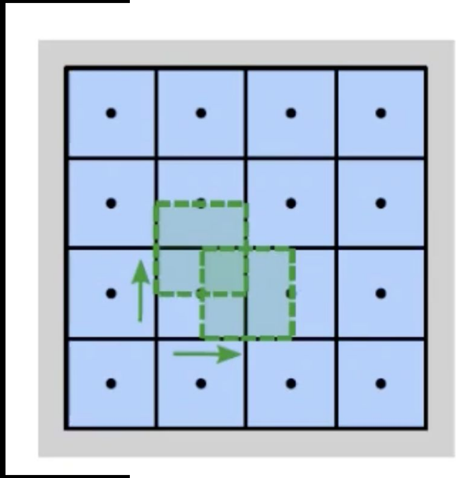
Types of grids

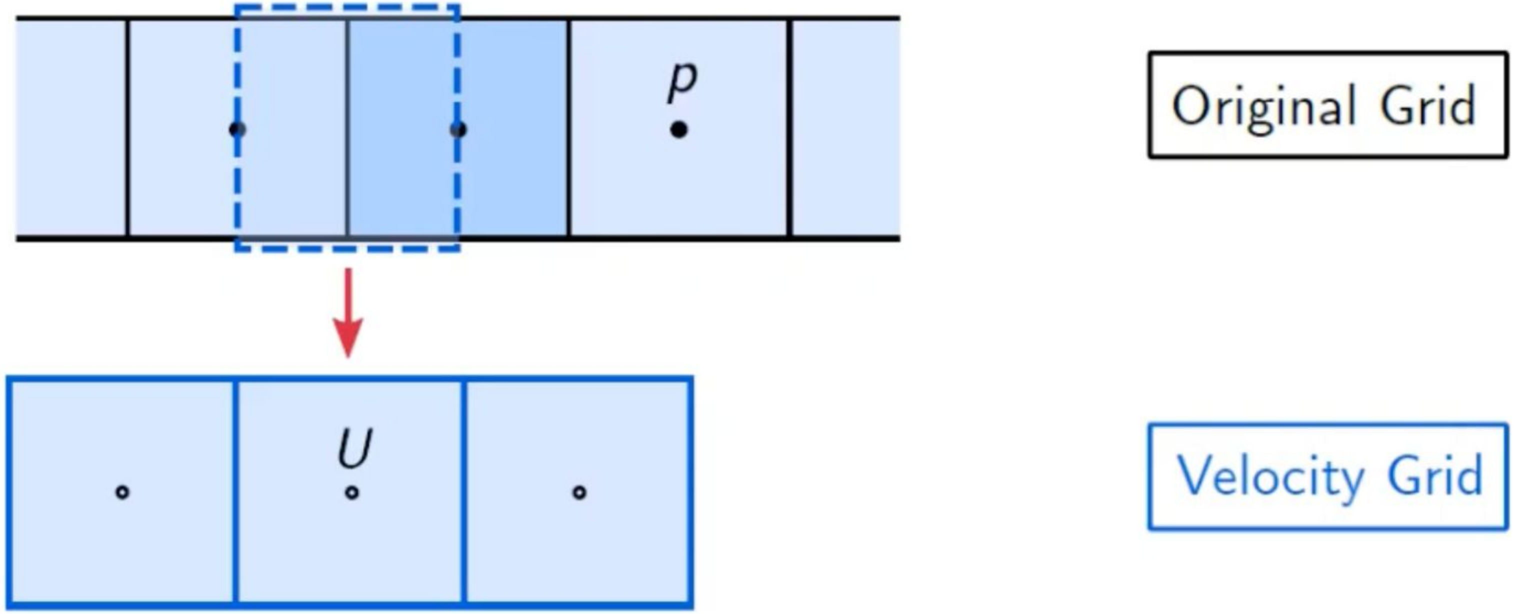


The previous solver used a collocated grid. Let's explore the staggered grid, it might solve the problem.

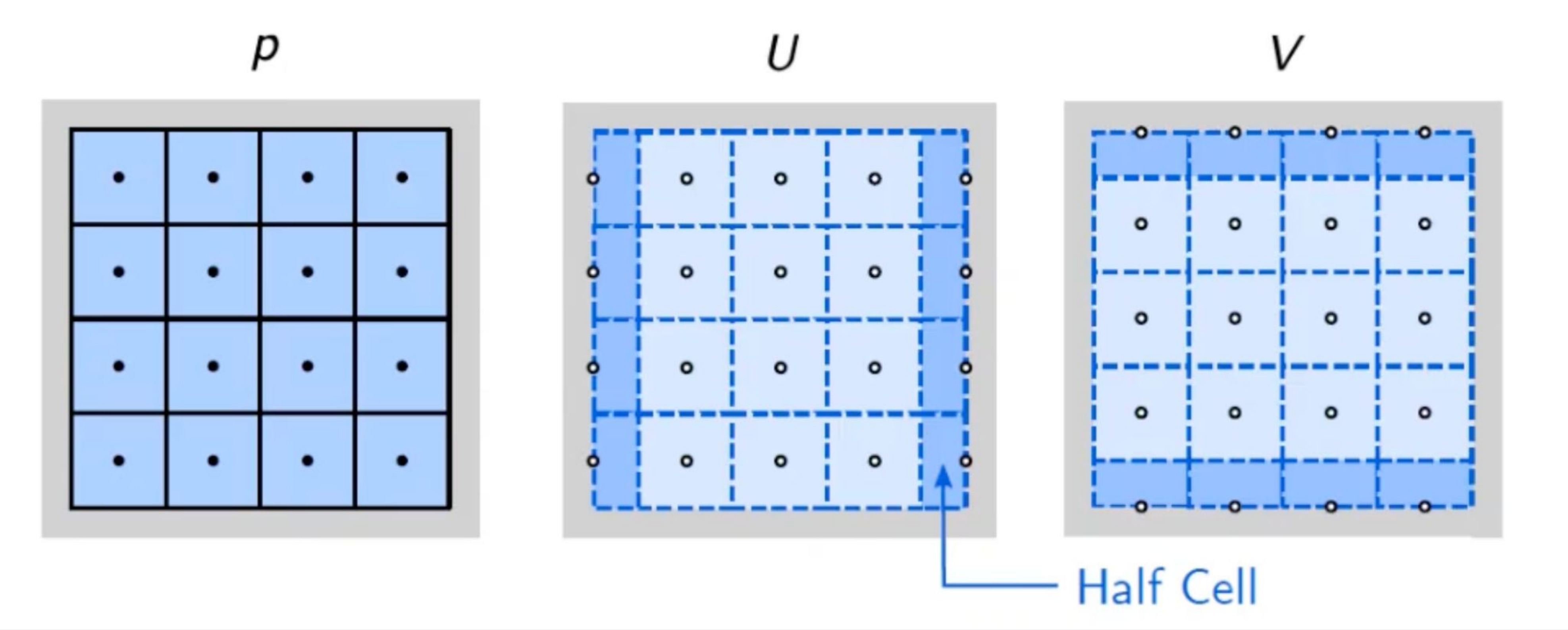
Staggered grids

- In a staggered grid, all flow variables are stored at the centroid, except for **velocity**.
- Velocity is calculated and stored on the **cell faces**.



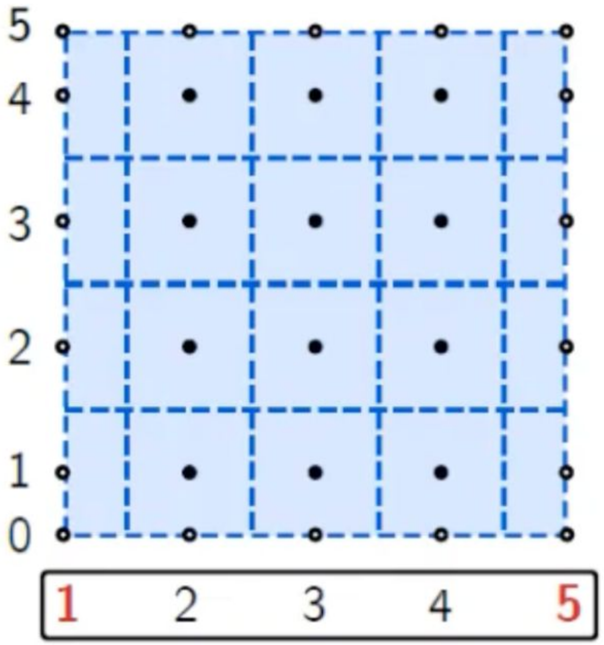


2D staggered grid

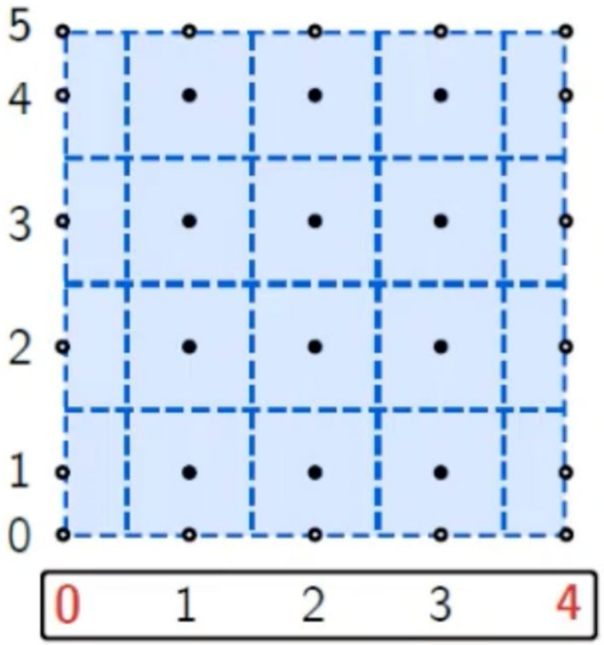


In 3D, we got 3 grids (all variables, u and v).

On a **staggered grid**, we have to choose between *forward* staggering and *backwards* staggering because of the half cells



(a) Forwards



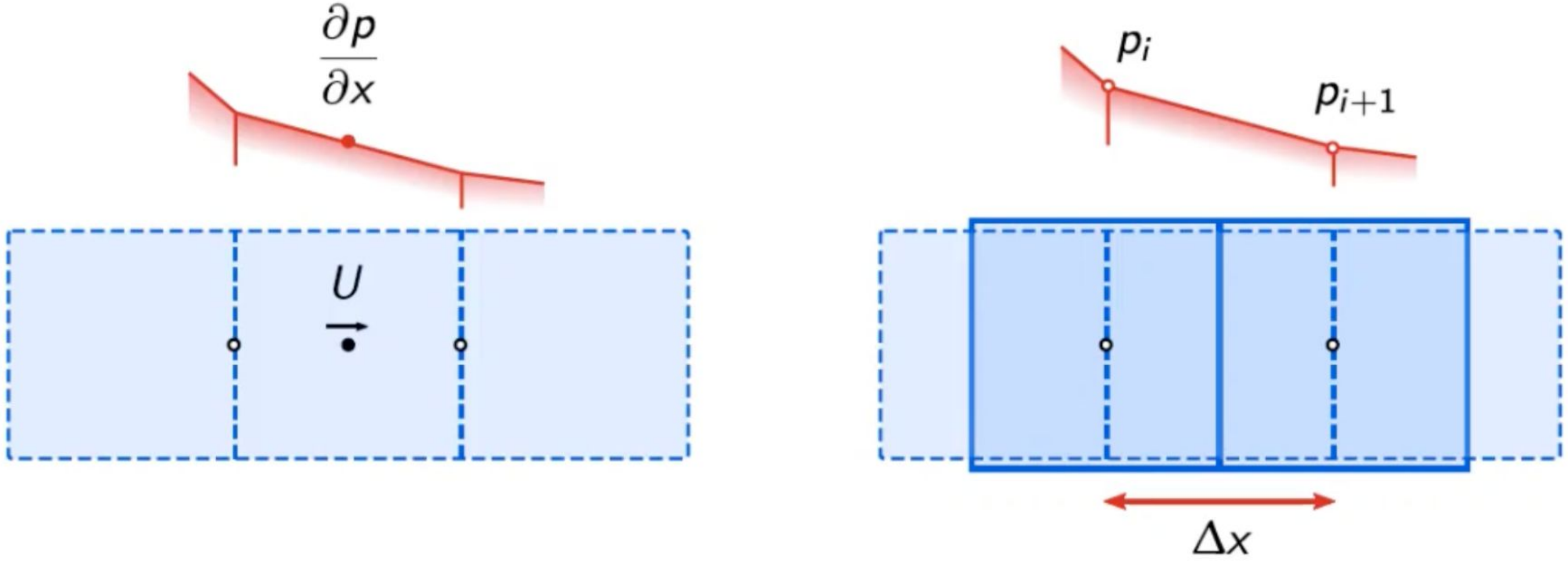
(b) Backwards

We need extra u's on the top and bottom boundaries and extra v's on left and right.
Here, they directly set the boundaries instead of using ghost points. Is it practically the same?

Pressure grid

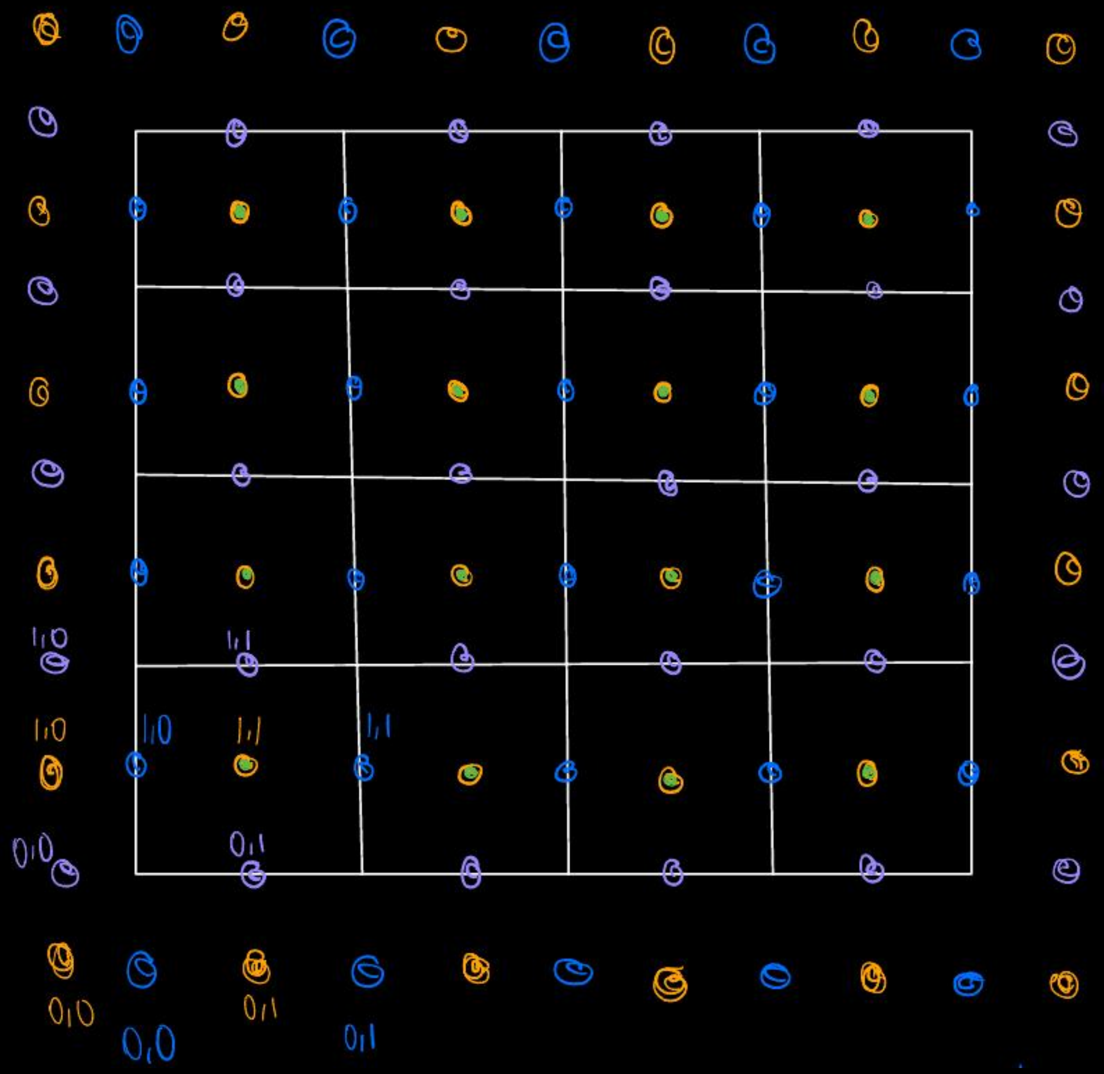
Pressure Gradient

- The pressure gradient across the U cell is calculated from the pressure at the centroids on the *original* grid.



$$\frac{\partial p}{\partial x} = \frac{p_{i+1} - p_i}{\Delta x}$$

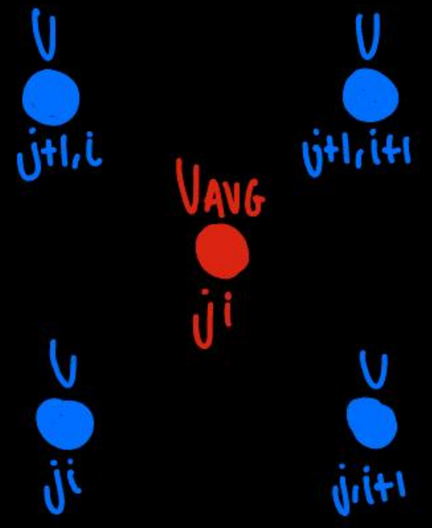
My staggered grid

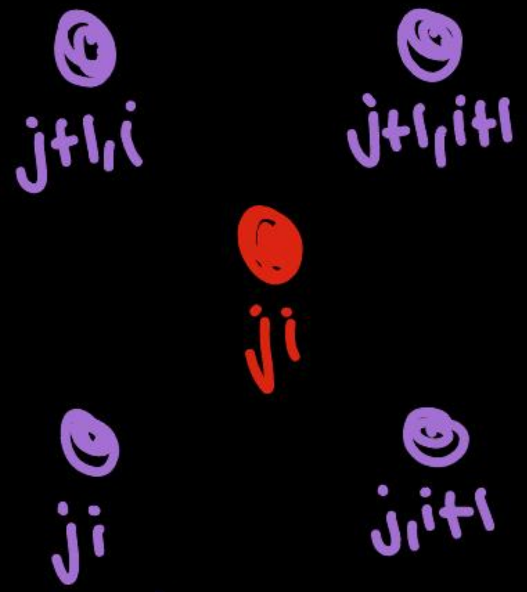


- U-velocity (N+2, N+1)
- V- velocity (N+1, N+2)
- p (N+2, N+2)
- b-term (N, N)

Interpolated velocities

- To compute du/dy and dv/dx, we’ll need to know the u-velocities at the v-positions and vice versa.
- Average velocities u_avg and v_avg are calculated using linear interpolation, assuming square cells.
- Warning!!!!! This method is exclusively designed for square-cell grids!



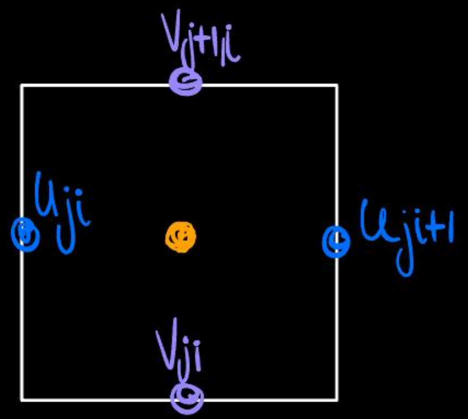


$$U_{AVG\,ji} = \frac{U_{ji} + U_{j+1,i} + U_{j+1,i+1} + U_{j,i+1}}{4}$$

$$V_{avg\,ji} = \frac{V_{ji} + V_{j,i+1} + V_{j+1,i} + V_{j+1,i+1}}{4}$$

Divergence

$$Div_{ji} = \frac{u_{j,i+1} - u_{j,i}}{dx} + \frac{V_{j+1,i} - V_{j,i}}{dy}$$



- Since it is a gradient, the divergence is associated to the center point.
- Units are [1/s].

Poisson solver: b-term, poisson, algorithm.

$$\underset{1}{\boxed{\frac{p^n_{i+1,j} - 2p^n_{i,j} + p^n_{i-1,j}}{\Delta x^2}}} + \underset{2}{\boxed{\frac{p^n_{i,j+1} - 2p^n_{i,j} + p^n_{i,j-1}}{\Delta y^2}}} =$$

$$\varrho\left[\frac{1}{\Delta t}\left(\frac{u_{i+1,j}-u_{i-1,j}}{2\Delta x} + \frac{v_{i,j+1}-v_{i,j-1}}{2\Delta y}\right) - \frac{u_{i+1,j}-u_{i-1,j}}{2\Delta x}\frac{u_{i+1,j}-u_{i-1,j}}{2\Delta x} - 2\frac{u_{i,j+1}-u_{i,j-1}}{2\Delta y}\frac{v_{i+1,j}-v_{i-1,j}}{2\Delta x} - \frac{v_{i,j+1}-v_{i,j-1}}{2\Delta y}\frac{v_{i,j+1}-v_{i,j-1}}{2\Delta y}\right]$$

Rearrange

Spans 2 cells cuz it comes from 2nd order central difference

$$p^n_{i,j} = \boxed{\frac{\left(p^n_{i+1,j} + p^n_{i-1,j}\right)\Delta y^2 + \left(p^n_{i,j+1} + p^n_{i,j-1}\right)\Delta x^2}{2\left(\Delta x^2 + \Delta y^2\right)}}$$

$$- \frac{\varrho \Delta x^2 \Delta y^2}{2\left(\Delta x^2 + \Delta y^2\right)}$$

B-term

$$\boxed{\times\left[\frac{1}{\Delta t}\left(\frac{u_{i+1,j}-u_{i-1,j}}{2\Delta x} + \frac{v_{i,j+1}-v_{i,j-1}}{2\Delta y}\right) - \frac{u_{i+1,j}-u_{i-1,j}}{2\Delta x}\frac{u_{i+1,j}-u_{i-1,j}}{2\Delta x} - 2\frac{u_{i,j+1}-u_{i,j-1}}{2\Delta y}\frac{v_{i+1,j}-v_{i-1,j}}{2\Delta x} - \frac{v_{i,j+1}-v_{i,j-1}}{2\Delta y}\frac{v_{i,j+1}-v_{i,j-1}}{2\Delta y}\right]}$$

$$\left.\frac{du}{dx}\right|_i = \frac{U_{i+1}-U_i}{dx}$$

U_{20} b_{10} U_{21}

dx

$$\left.\frac{dv}{dy}\right|_{ji} = \frac{V_{j+1}-V_j}{dy}$$

V_{j+1} b_j V_j

dy

$$\left.\frac{du}{dy}\right|_j = \frac{U_{AVG\,j+1}-U_{AVG\,j}}{dy}$$

$U_{AVG\,j+1}$ b_j $U_{AVG\,j}$

dy

$$\left.\frac{dv}{dx}\right|_i = \frac{V_{j,i+1}-V_{j,i}}{dx}$$

$V_{avg\,i}$ b_i $V_{AVG\,i}$

dx

1

$$\left.\frac{dp}{dx}\right|_i = \frac{p_{i+1}-p_{i}}{2\cdot dx}$$

2·dx

P_{i-1} b_i P_{i+1}

2

$$\left.\frac{dp}{dy}\right|_j = \frac{P_{j+1}-P_{j-1}}{2\cdot dy}$$

P_{j+1} b_j P_{j-1}

2dy

Poisson solver: b-term, poisson, algorithm.

$$\frac{p^n_{j,i+1} - 2p^n_{ji} + p^n_{j,i-1}}{\Delta x^2} + \frac{p^n_{j+1,i} - 2p^n_{ji} + p^n_{j-1,i}}{\Delta y^2} = \rho\left[\frac{1}{\Delta t}\left(\frac{u_{j,i+1}-u_{j,i}}{\Delta x} + \frac{v_{j+1,i}-v_{ji}}{\Delta y}\right) - \left(\frac{u_{j,i+1}-v_{ji}}{2\Delta x}\right)^2 - 2\,\frac{u_{AVG\,j+1,i}-u_{AVG\,ji}}{\Delta y}\cdot\frac{v_{AVG\,j,i+1}-v_{AVG\,ji}}{\Delta x} - \left(\frac{v_{j+1,i}-v_{ji}}{\Delta y}\right)^2\right]$$

explicit

$$p^n_{ji} = \frac{\left(p^n_{j,i+1}+p^n_{j,i-1}\right)\cdot\Delta y^2 + \left(p^n_{j+1,i}+p^n_{j-1,i}\right)\cdot\Delta x^2}{2\left(\Delta x^2+\Delta y^2\right)} - \frac{\rho\Delta x^2\Delta y^2}{2\left(\Delta x^2+\Delta y^2\right)}\cdot\left[\frac{1}{\Delta t}\left(\frac{u_{j,i+1}-u_{ji}}{\Delta x}+\frac{v_{j+1,i}-v_{ji}}{\Delta y}\right) - \left(\frac{u_{ji+1}-u_{ji}}{\Delta x}\right)^2 - 2\cdot\frac{u_{AVG\,j+1,i}-u_{AVG\,ji}}{\Delta y}\cdot\frac{v_{AVG\,j,i+1}-v_{AVG\,ji}}{\Delta x} - \left(\frac{v_{j+1,i}-v_{ji}}{\Delta y}\right)^2\right]$$

$$\left.\frac{du}{dx}\right|_i = \frac{u_{i+1}-u_i}{dx}$$

U_{20} b_{10} U_{21}

dx

$$\left.\frac{dv}{dy}\right|_{ji} = \frac{v_{j+1}-v_j}{dy}$$

v_{j+1}

dy b_j

v_j

$$\left.\frac{dv}{dx}\right|_i = \frac{v_{j,i+1}-v_{ji}}{dx}$$

$v_{avg\,i}$ b_i $v_{AVG\,i+1}$

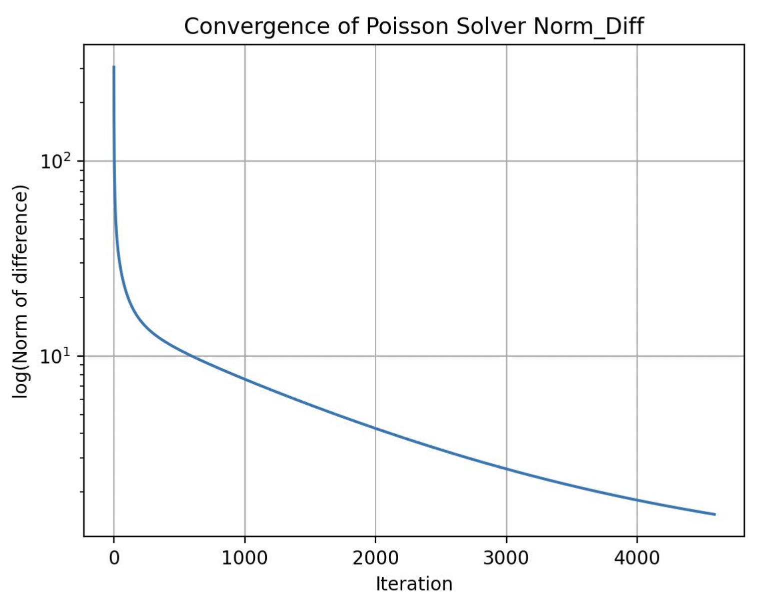
dx

$$\left.\frac{du}{dy}\right|_j = \frac{u_{AVG\,j+1}-u_{AVG\,j}}{dy}$$

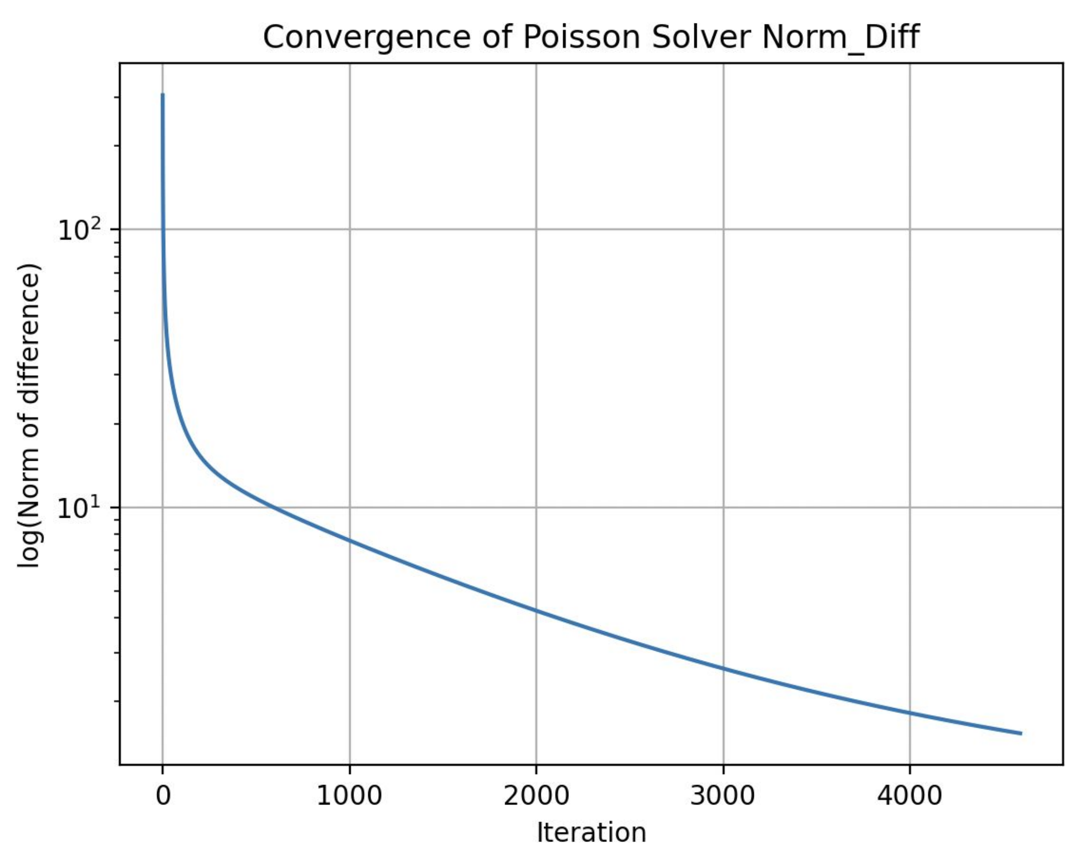
$u_{AVG\,j+1}$

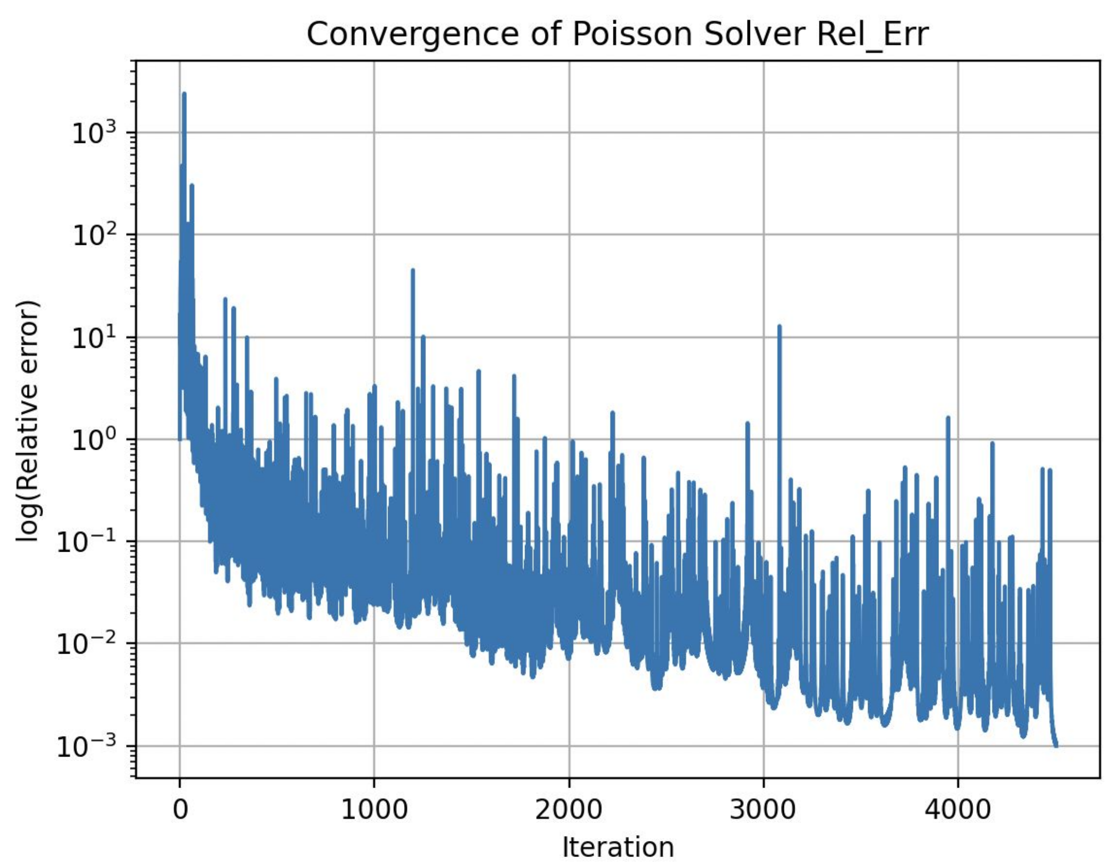
dy b_j

$u_{AVG\,j}$



Poisson solver: Convergence test





*Iterative resolution of the poisson equation using a randomly initialized velocities on a 80x80 grid.
*The norm of difference and relative error are computed in between two consecutive pressure arrays.
Why does the relative error have so much noise but the norm of difference doesn't? Because rel_err is relative (way more variability) and norm_diff is absolute.

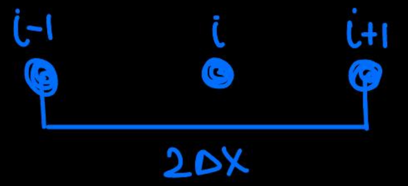
Staggered Discretization of Advection-Diffusion: U-velocity

$$\frac{U_{ji}^{n+1}-U_{ji}^{n}}{\Delta t} + U_{ji}^{n}\cdot\frac{U_{j,i+1}^{n}-U_{j,i-1}^{n}}{2\cdot\Delta x} + V_{avg\,ji}\cdot\frac{U_{j+1,i}^{n}-U_{j-1,i}^{n}}{2\Delta y} - \frac{1}{\rho}\cdot\frac{P_{j,i+1}^{n}-P_{ji}^{n}}{\Delta x} + \nu\left(\frac{U_{j,i+1}^{n}-2U_{ji}^{n}+U_{j,i-1}}{2\cdot\Delta x} + \frac{U_{j+1,i}^{n}-2\cdot U_{ji}^{n}+U_{j-1,i}^{n}}{2\cdot\Delta y}\right)$$

explicit

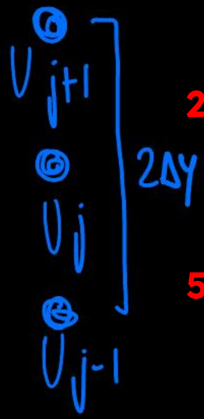
$$U_{ji}^{n+1} = U_{ji}^{n} - U_{ji}^{n}\cdot\Delta t\cdot\underbrace{\frac{U_{j,i+1}^{n}-U_{j,i-1}^{n}}{2\Delta x}}_{1} - V_{AVG\,ji}\cdot\Delta t\,\underbrace{\frac{U_{j+1,i}^{n}-U_{j-1,i}^{n}}{2\Delta y}}_{2} - \frac{\Delta t}{\rho}\cdot\underbrace{\frac{P_{j,i+1}-P_{ji}}{\Delta x}}_{3} + \nu\cdot\Delta t\left(\underbrace{\frac{U_{j,i+1}^{n}-2U_{ji}^{n}+U_{j,i-1}^{n}}{\Delta x^2}}_{4} + \underbrace{\frac{U_{j+1,i}^{n}-2U_{ji}^{n}+U_{j-1,i}^{n}}{\Delta y^2}}_{5}\right)$$

3 $$\left.\frac{dp}{dx}\right|_i = \frac{P_{i+1}-P_i}{\Delta x}$$



1 $$\left.\frac{du}{dx}\right|_i = \frac{U_{i+1}-U_{i-1}}{2\Delta x}$$

4 $$\left.\frac{d^2u}{dx^2}\right|_i = \frac{U_{i+1}-2U_i+U_{i-1}}{\Delta x^2}$$

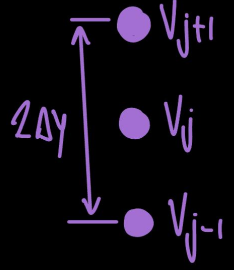


2 $$\left.\frac{du}{dy}\right|_j = \frac{U_{j+1}-U_{j-1}}{2\Delta y}$$

5 $$\left.\frac{d^2u}{dy^2}\right|_j = \frac{U_{j+1}-2U_j+U_{j-1}}{\Delta y^2}$$

Staggered Discretization of Advection Diffusion: V-velocity

$$\frac{V_{ji}^{n+1}-V_{ji}^{n}}{\Delta t} + u_{AVG,ji}^{n}\cdot\frac{V_{j,i+1}^{n}-V_{j,i-1}^{n}}{2\cdot\Delta x} + V_{ji}^{n}\cdot\frac{V_{j+1,i}^{n}-V_{j-1,i}^{n}}{2\cdot\Delta y} = -\frac{1}{\rho}\cdot\frac{P_{j+1,i}-P_{j,i}}{\Delta y} + \nu\left(\frac{V_{j,i+1}^{n}-2V_{j,i}^{n}+V_{j,i-1}^{n}}{\Delta x^2} + \frac{V_{j+1,i}^{n}-2V_{ji}^{n}+V_{j-1,i}^{n}}{\Delta y^2}\right)$$



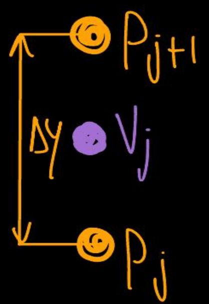
2 $\left.\frac{dv}{dy}\right|_j = \frac{V_{j+1}-V_{j-1}}{2\Delta y}$

5 $\left.\frac{d^2v}{dy^2}\right|_j = \frac{V_{j+1}-2V_j+V_{j-1}}{\Delta y^2}$

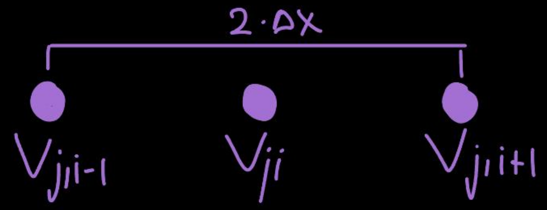
Explicit form

$$V_{ji}^{n+1} = V_{ji}^{n} - u_{AVG,ji}^{n}\cdot\frac{\Delta t}{2\Delta x}\cdot\overbrace{\left(V_{j,i+1}^{n}-V_{j,i-1}^{n}\right)}^{1} - V_{ji}^{n}\cdot\frac{\Delta t}{2\Delta y}\overbrace{\left(V_{j+1,i}^{n}-V_{j-1,i}^{n}\right)}^{2} - \frac{\Delta t}{\rho\,\Delta y}\underbrace{\left(P_{j+1,i}-P_{ji}\right)}_{3} + \nu\cdot\Delta t\left(\underbrace{\frac{V_{j,i+1}^{n}-2V_{j,i}^{n}+V_{j,i-1}^{n}}{\Delta x^2}}_{4} + \underbrace{\frac{V_{j+1,i}^{n}-2V_{ji}^{n}+V_{j-1,i}^{n}}{\Delta y^2}}_{5}\right)$$

3



$\left.\frac{dp}{dy}\right|_j = \frac{P_{j+1}-P_j}{\Delta y}$



1 $\left.\frac{dv}{dx}\right|_i = \frac{V_{i+1}-V_{i-1}}{2\cdot\Delta x}$

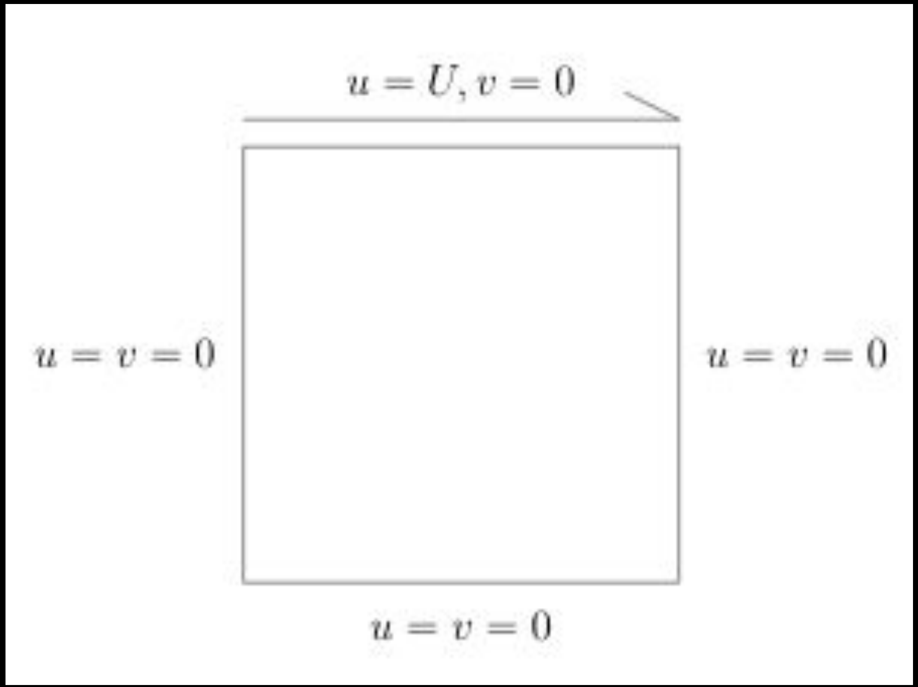
4 $\left.\frac{d^2v}{dx^2}\right|_i = \frac{V_{i+1}-2V_i+V_{i-1}}{\Delta x^2}$

Cavity with moving Lid Simulation set-up

- Square domain
- 2D
- Uniform cartesian grid made out of square cells

Boundary conditions:

- Velocity -> No Slip condition an all 4 walls:
  - Left, Bottom, Right walls: $\underline{u}$=<0,0>
  - Top wall (moving lid): $\underline{u}$=<Ulid,0>
- Pressure -> Neumann:
  - Zero gradient at all 4 walls.



The Algorithm: Predictor Corrector (Fractional Step Method)

We couldn't get rid of divergence, so we'll try this new algorithm, that is the one used by Matthias Muller.

1

Solve Momentum

Neglect pressure gradient term

2

Solve Poisson equation

Jacobi solver
(Gauss-Seidel could be way faster)

Root mean square error (RMS)
Threshold value.

3

Correct velocities

Apply the pressure gradient term.

Ensures a divergence free flow

Other Updates

- Everytime we modify velocities we update the BCs. Because the way they are defined now, they are affected by the interior points.
- We introduce a conservative dynamic timestep, taking into account advection and diffusivity:

$$① \; C_{ADV} \longrightarrow dt = \frac{cfl \cdot dx}{u_{MAX}} \quad \text{(Updated at every step)}$$

$$② \; C_{DIFF} \longrightarrow dt = \frac{C}{2\nu\left(\frac{1}{dx^2} + \frac{1}{dy^2}\right)} \quad \text{(Constant)}$$

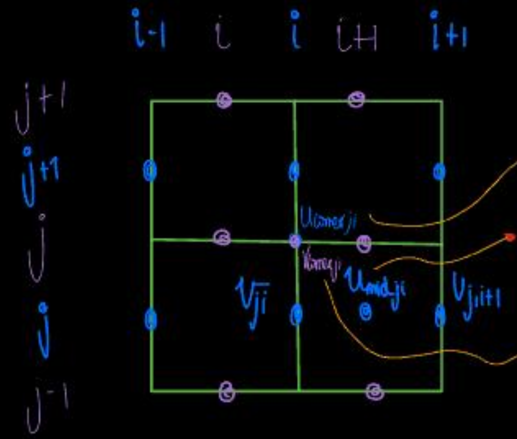
$\rightarrow$ Ensure stability

- Discretization on advection points is performed using the interpolated values in the corners (see next slide).

New advection approach (conservative form)

$$U^{n+1}_{ji} = U^{n}_{ji} + dt \cdot \Bigg[nu \cdot \Big(\underbrace{\frac{d^2u}{dx^2} + \frac{d^2u}{dy^2}}_{\text{Diffusion}} \Big) - \underbrace{\frac{d}{dx}u^2}_{\text{u-advect}} - \underbrace{\frac{d}{dy} \cdot uv}_{\text{v-advect}} \Bigg]$$

$$\frac{d}{dx}u^2 \rightarrow \frac{u_{i+1}^2 - u_i^2}{dx}$$



$$V_{corner_j} = \frac{V_{j+1} - V_j}{2}$$

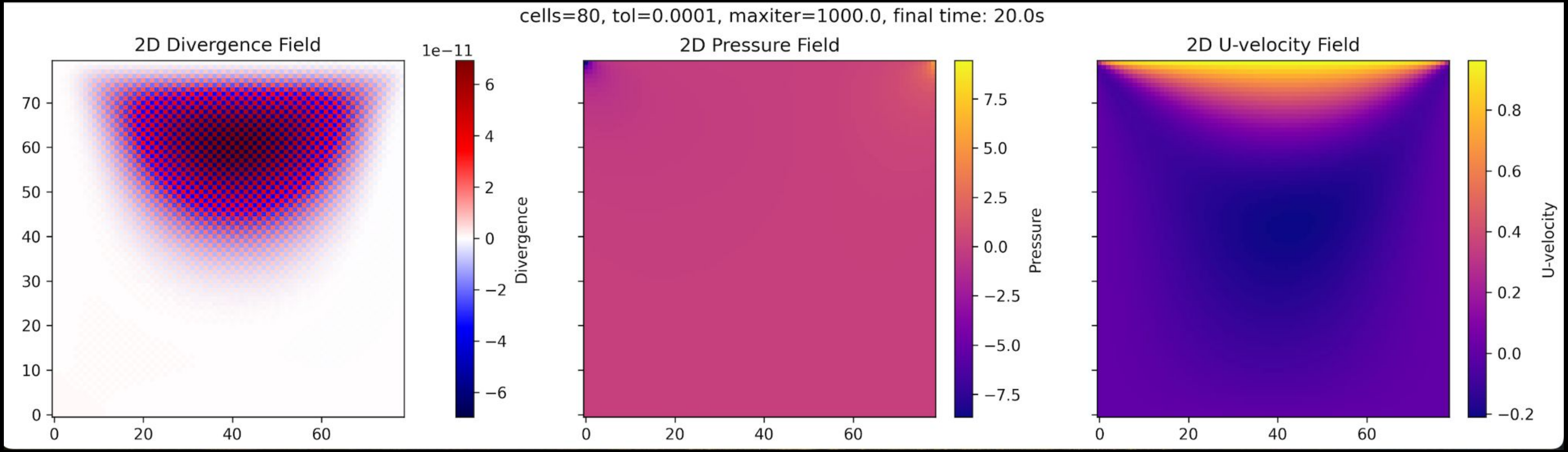
$$U_{mid_i} = \frac{U_{i+1} + U_i}{2}$$

$$V_{corner} = \frac{V_{j+1} + V_j}{2}$$

$$\left.\frac{d(u^2)}{dx}\right|_i = \frac{U_{mid,i+1}^2 - U_{mid,i}^2}{dx}$$

$$\left.\frac{d(u \cdot v)}{dy}\right|_i = \frac{\left(V_{corner\,ji} \cdot V_{corner\,ji} - U_{corner\,j-1,i} \cdot V_{corner\,j-1,i}\right)}{dy}$$

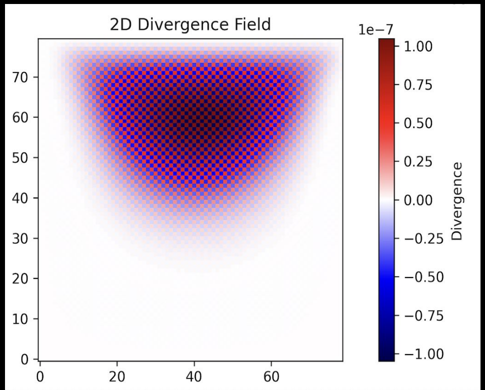
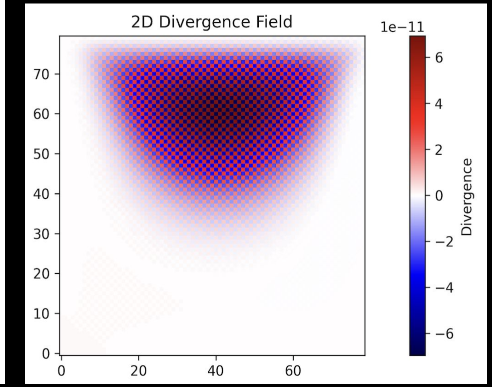
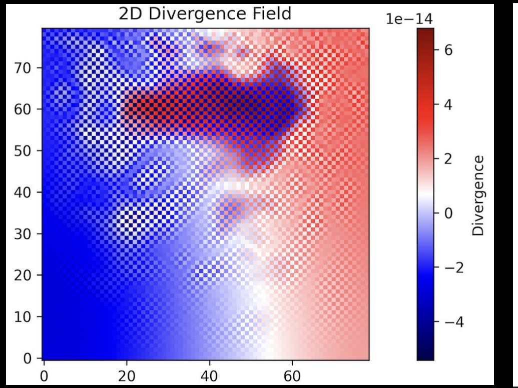
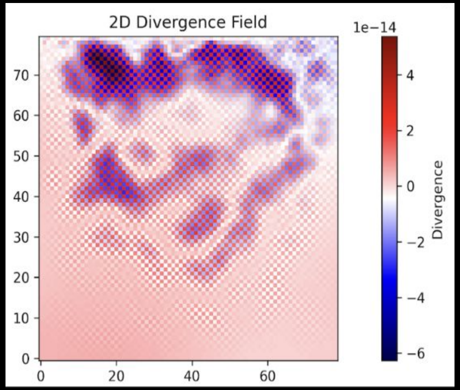
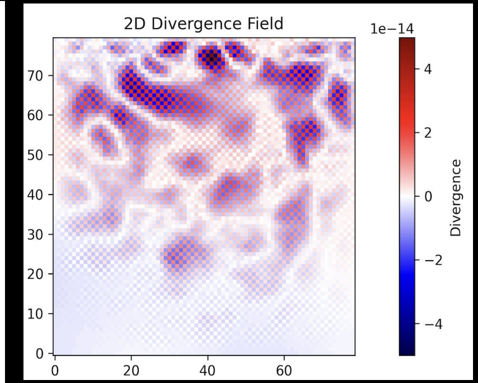
Solutions



cells=80, tol=0.0001, maxiter=1000.0, final time: 20.0s
2D Divergence Field
1e−11
Divergence
2D Pressure Field
Pressure
2D U-velocity Field
U-velocity

How does the divergence evolves for different Poisson tolerance?

Tolerance

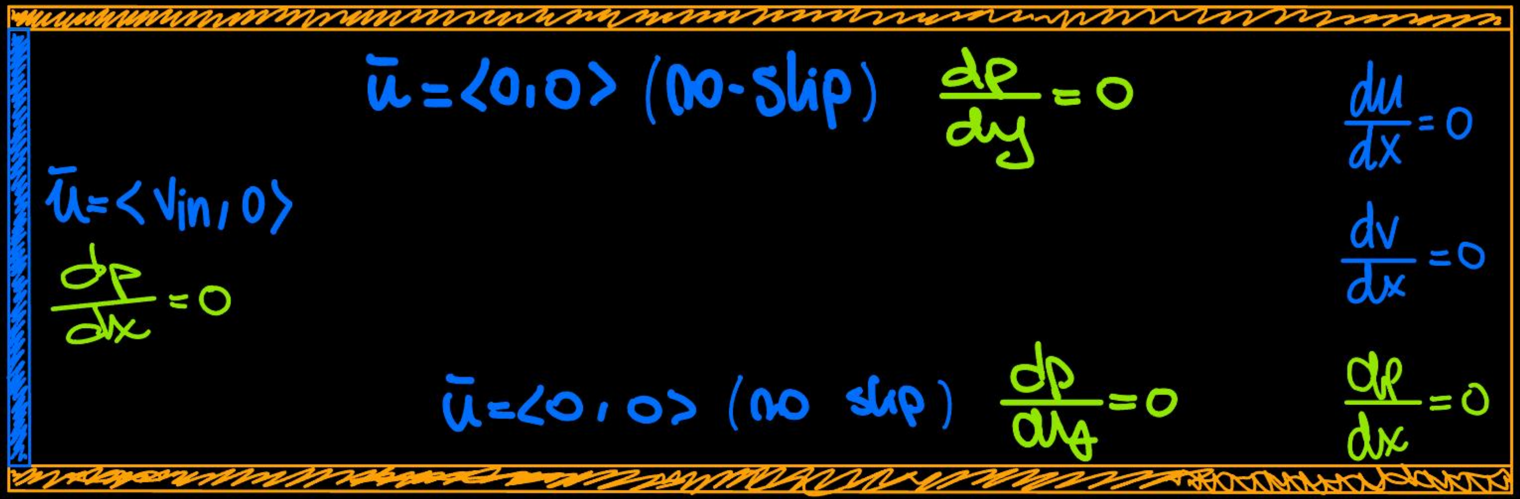
1e-3	1e-4	1e-5	1e-6	1e-7
				

Would be interesting to indicate how long it took to run, even though this is a future worry.

Channel Flow set-up

BC's:

- Inlet:
  - u=2, v=0
  - dP/dx=0
- Outlet:
  - du/dx=0, dv/dx=0
  - dp/dx=0
- Top and bottom:
  - u=0, v=0 (non-slip)
  - dp/dy=0

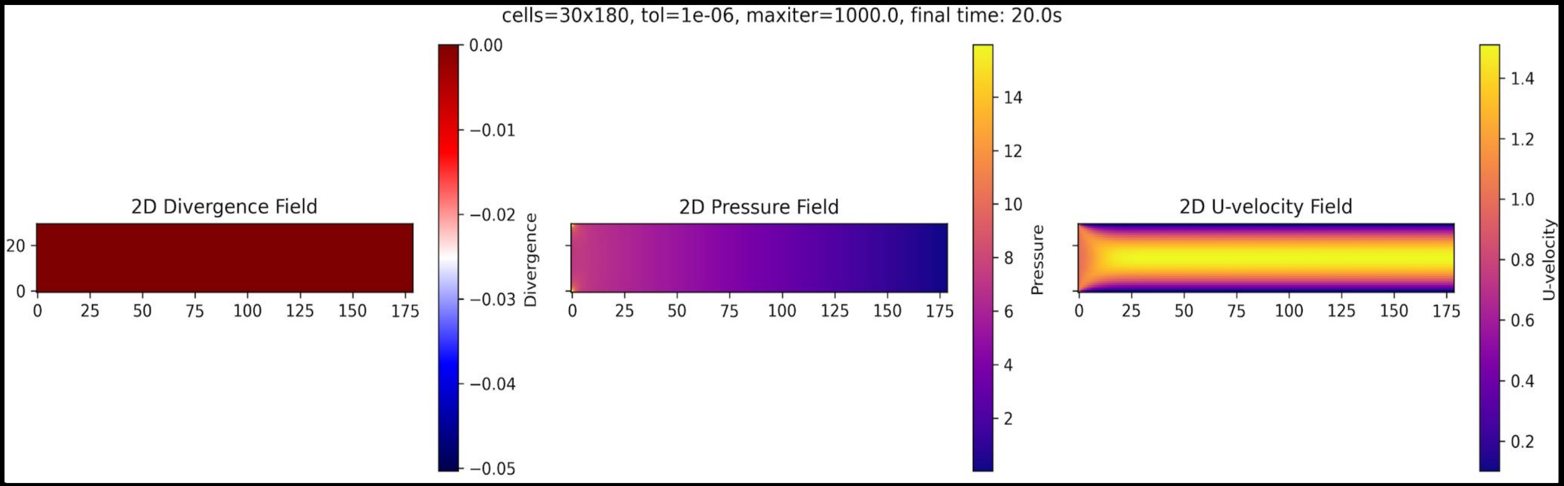


It's running super slow compared with the same domain and resolution with previous BCs. Why?

Channel flow - Solutions

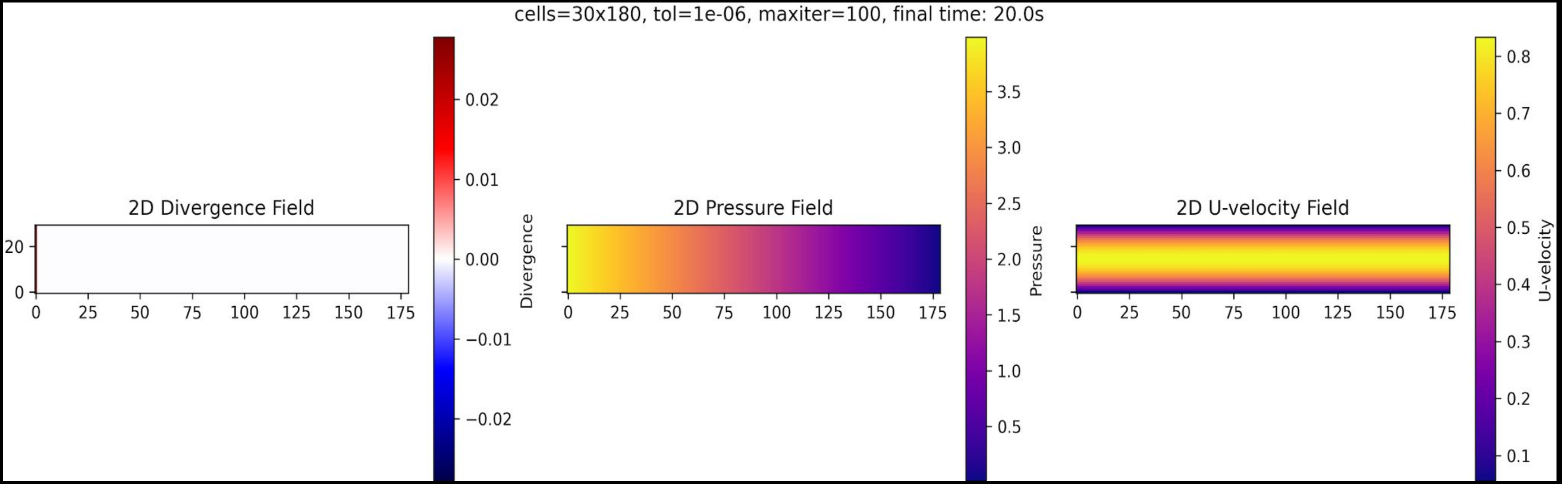
BC's:

- Velocities.
  - In: invel u=<1,0>
  - Out: du/dx=dv/dx=0
  - Top & bottom: Non-slip u=<0,0>
- Pressure.
  - Top & bottom: dp/dy=0
  - Inlet: dp/dx=0
  - Outlet: p=0



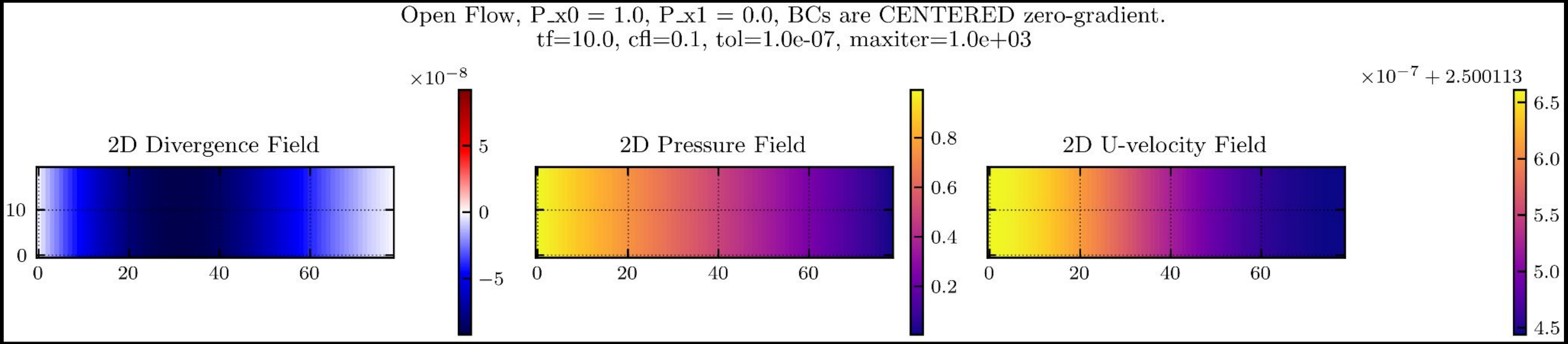
BC's

- Velocities
  - In & Out: du/dx=dv/dx=0
  - Top & bottom: Non-slip u=<0,0>
- Pressure
  - In: p=4
  - Out: p=0
  - Top & bottom: dp/dy=0



Comments: Both cases are divergence free except of at the inlet (or maybe outlet)
A way to solve this could be adding u-vel ghost nodes for properly setting du/dx at x=0.

Channel flow with extra nodes - Solutions



BC's:

- Velocities.
  - In & Out: du/dx=dv/dx=0
  - Top & bottom: du/dy=dv/dy=0
- Pressure.
  - Top & bottom: dp/dy=0
  - Inlet: p=1
  - Outlet: p=0

What is new?

- New ghost cells -> Allows defining BCs at the actual boundary.

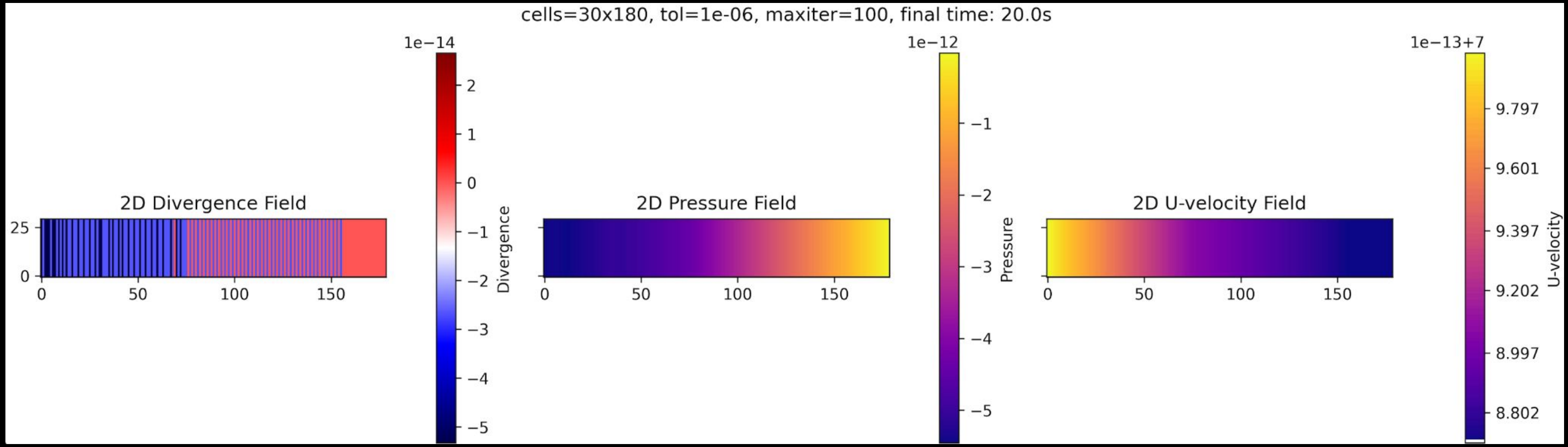
Comments

This solves the problem of the non-zero divergence at the outlet.

Open field - Solutions

BC’s:

- Velocities.
  - In: invel u=<7,0>
  - Out: du/dx=dv/dx=0
  - Top & bottom: du/dy=dv/dy=0
- Pressure.
  - Top & bottom: dp/dy=0
  - Inlet: dp/dx=0
  - Outlet: p=0



BC’s:

- Velocities.
  - In: invel u=<7,0>
  - Out: du/dx=dv/dx=0
  - Top & bottom: du/dy=dv/dy=0
- Pressure.
  - Top & bottom: dp/dy=0
  - Inlet: dp/dx=0
  - Outlet: dp/dx=0