

Fluid solver: theory notes

Free of cost. Notes: CC BY 4.0. Code: MIT. DRAFT: licences pending confirmation by the author.

Contents

1	Semi-Lagrangian advection never blows up, but every step blurs: the interpolation is a hidden diffusion	1
2	Jacobi iteration smooths the pressure but converges at a rate $\cos(\pi/(N+1))$: 6 to 24 sweeps are nowhere near a solve	8
3	Incompressible flow: the velocity must have zero divergence, and the pressure is whatever enforces it	15
4	The projection solves one sparse matrix every step: minus four on the diagonal, plus one for each neighbour	22
5	Vorticity confinement and buoyancy are forces added by hand: the first puts back the curls the grid smears out, the second makes hot gas rise	29

1 Semi-Lagrangian advection never blows up, but every step blurs: the interpolation is a hidden diffusion

Stam's advection traces each cell back along the velocity and copies the value found there. It is stable for any time step because it only averages, but linear interpolation averages neighbours and so smears sharp features exactly like a diffusion with a known coefficient. MacCormack's correction removes most of the blur and a clamp keeps it from overshooting.

Read the figure

Question. How sharp is the interface after the fluid has been moved around for a while, and how much of the smoothing is numerical?

Axes. The picture is the simulation box, x to the right and y up. Colour is the passive dye: teal where the dye is, dark where it is not. The dye does not act on the flow; it only makes the motion visible.

Look at.

- The width of the edge between teal and dark: it should be a few cells, not tens of cells.
- Whether the spirals are crisp or have dissolved: numerical diffusion dissolves them first.
- Overshoot: dye brighter than the original teal or darker than none would be a sign of a scheme without a clamp.

Must check.

- Grid size and time step (the blur per step depends on both).

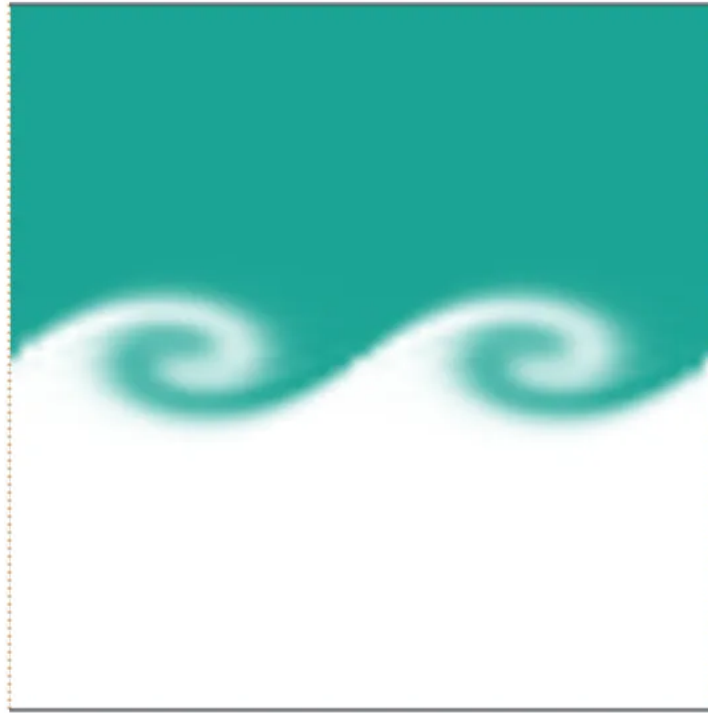


Figure 1: Screenshot of the Flow workspace (scene 1, dye view) after 1.7 s: the Kelvin-Helmholtz roll-up of a shear layer, advected with MacCormack.

- The advection scheme (semi-Lagrangian or MacCormack).
- Whether a clamp was applied.

If it is wrong.

- Reading a smooth blob as physical mixing when it is the interpolation's diffusion.
- Comparing two runs with different grids or time steps and calling the sharper one more accurate without checking mass.

What it proves. That the scheme keeps fine structure for the simulated time; the sharpness belongs to the scheme and the grid, not to the physics, because the dye here has no diffusion of its own.

Terms

Advection $\partial_t q + u \cdot \nabla q = 0$ Transport of a quantity by the flow, with no change to the quantity itself: the quantity is constant along a path of the fluid.

Semi-Lagrangian step $q^{n+1}(x) = q^n(x - \Delta t u(x))$ For each grid cell, trace the path backward over one time step, read the old field there by interpolation, and write it into the cell.

Interpolation weight $\alpha = \frac{u \Delta t}{h} - \lfloor \frac{u \Delta t}{h} \rfloor$ The fractional part of the shift in cells. Linear interpolation mixes the two neighbouring cells with weights $1 - \alpha$ and α .

CFL number $\text{CFL} = \frac{|u| \Delta t}{h}$ How many cells the flow crosses in one step. Explicit upwind schemes need it at most 1; the semi-Lagrangian step has no such limit but loses accuracy when it exceeds 1.

Numerical diffusion ν_{num} Smoothing that comes from the scheme, not from the equation. It acts like a diffusion with a coefficient that depends on h , the shift and the time step.

Amplification factor $g(\theta)$ The factor by which one step multiplies a Fourier mode. A scheme is stable when it never exceeds 1 in size.

MacCormack correction $q^{n+1} = \hat{q} + \frac{1}{2}(q^n - \hat{q})$ Advect forward, advect the result backward, and add half of the difference from the start: a second-order estimate of the error to cancel.

Limiter (clamp) $\min \leq q \leq \max$ Restricting the corrected value to the range of the four cells it was interpolated from, so that no new maxima or minima are created.

Remember

- Semi-Lagrangian: $q_i^{n+1} = (1 - \alpha) q_{i-m}^n + \alpha q_{i-m-1}^n$ for a shift of $m + \alpha$ cells. Always a convex combination, so $\max |q|$ never grows.
- Fourier mode $e^{i\theta j}$: $|g|^2 = 1 - 2\alpha(1 - \alpha)(1 - \cos \theta) \leq 1$. A two-cell wave ($\theta = \pi$) is multiplied by $|1 - 2\alpha|$: at $\alpha = \frac{1}{2}$ it vanishes in one step.
- Variance of a smeared profile grows by $\alpha(1 - \alpha)h^2$ per step, which is a diffusion $\nu_{\text{num}} = \alpha(1 - \alpha)h^2/(2\Delta t)$.
- The blur does not depend on the whole shift but only on its fractional part α , and it is zero for a whole-cell shift.
- MacCormack with the clamp is second order where the field is smooth and falls back to a first-order step at extrema.

Understand

Easy: copying from where you came from Imagine each cell asking: where was the water that is here now one moment ago? The answer is a point between cells, so the cell takes a blend of its neighbours' old values. That never makes a number bigger than anything it blended, so nothing explodes. But blending sharp edges rounds them off, every step, and many steps smear a crisp line into a soft ramp.

Medium: the blend is a diffusion Shift a profile by half a cell with linear interpolation: each new value is half of one old cell plus half of the next. A spike of height one becomes two cells of one half. The spread, measured as variance, grows by $\alpha(1 - \alpha)h^2 = \frac{1}{4}h^2$ per step. That is the spread of a diffusion with coefficient 0.000814 (lengths squared per second) for a 96-cell grid and a step of 1/60 s, about 41 % of the viscosity 0.002 of the cavity scene. The numerical diffusion is not small compared with the physical one.

Hard: stable, but not accurate; MacCormack buys the accuracy back The amplification factor of the linear-interpolation step never exceeds one in size for any α (a scan over α and θ gives 1), which is why the method is unconditionally stable. Running the real solver for 100 steps with a half-cell shift, a Gaussian of width 3 cells grew to 5.831 cells (theory 5.831) and its peak fell from 1 to 0.514. With MacCormack and the clamp the width was 3.53 cells and the peak 0.83. The price is that the mass is no longer exactly conserved: it rose by 1.0 % in that run.

Hardest: advection is also how the velocity damps itself The velocity is advected by itself, so the same blur acts on the momentum: it is an artificial viscosity that grows with h and has nothing to do with the physical ν . In a Kelvin-Helmholtz roll-up the blur thickens the shear layer and slows the instability; in a vortex street it can erase the street altogether. That is why vorticity confinement exists (next page): it puts back the small rotation that the interpolation took away. A scheme that is both stable and sharp needs to know where the field is smooth, which is what the clamp does in a crude way.

Derivation

Step 1. State the transport equation

$$\partial_t q + u \cdot \nabla q = 0 \iff \frac{dq}{dt} = 0 \text{ along } \frac{dx}{dt} = u(x, t)$$

q is constant along a path. The exact answer is q at the foot of the path.

Step 2. Semi-Lagrangian step

$$q^{n+1}(x) = q^n(x - \Delta t u(x))$$

One backward step of the path equation. The value is read from the old field by interpolation.

Step 3. Linear interpolation in one dimension

$$q_i^{n+1} = (1 - \alpha) q_{i-m}^n + \alpha q_{i-m-1}^n, \quad \frac{u \Delta t}{h} = m + \alpha$$

A convex combination: stable. All the blur sits in the fractional part α .

Step 4. Amplification factor of a Fourier mode

$$g(\theta) = (1 - \alpha) + \alpha e^{-i\theta}, \quad |g|^2 = 1 - 2\alpha(1 - \alpha)(1 - \cos \theta)$$

Insert $q_j = e^{ij\theta}$ and take the modulus squared. Never above 1, smallest for the shortest waves.

Step 5. Read off the numerical diffusion

$$\text{Var } q \text{ grows by } \alpha(1 - \alpha)h^2 \text{ per step} \Rightarrow \nu_{\text{num}} = \frac{\alpha(1 - \alpha)h^2}{2\Delta t}$$

A diffusion equation spreads variance by $2\nu t$. Equate the two.

Step 6. Correct it: MacCormack with a clamp

$$\hat{q} = \mathcal{A}_{\Delta t} q^n, \quad \hat{\hat{q}} = \mathcal{A}_{-\Delta t} \hat{q}, \quad q^{n+1} = \text{clamp}(\hat{\hat{q}} + \frac{1}{2}(q^n - \hat{\hat{q}}))$$

Going forward and back should return the start; the difference estimates the error. Half of it is added back, then the result is held inside the local range.

Worked example: How much does a half-cell shift blur a Gaussian in 100 steps?

A Gaussian of standard deviation 3 cells is advected at a constant speed that moves it half a cell per step ($\alpha = 1/2$), with semi-Lagrangian linear interpolation. What is its standard deviation after 100 steps, and what is the equivalent diffusion coefficient on a 96-cell grid with a step of 1/60 s?

1. Each step spreads the variance by $\alpha(1 - \alpha)$ cells squared.

$$\Delta\sigma^2 = \frac{1}{2} \cdot \frac{1}{2} = 0.25 \text{ cells}^2$$

2. After 100 steps the variance is the initial one plus 100 times that.

$$\sigma^2 = 9 + 100 \times 0.25 = 34$$

3. Take the square root.

$$\sigma = \sqrt{34} = 5.831 \text{ cells}$$

4. Equivalent diffusion: $2\nu_{\text{num}}t$ equals the added variance $25 h^2$ over $t = 100/60$ s, with $h = 1/96$.

$$\nu_{\text{num}} = \frac{0.25 h^2}{2 \Delta t} = \frac{0.25/9216}{2/60} = 8.14 \times 10^{-4}$$

Answer. $\sigma = 5.831$ cells, and $\nu_{\text{num}} = 0.000814$ lengths squared per second. The Flow solver, run for exactly this case, gave $\sigma = 5.831$.

Analyze

1. Why is the semi-Lagrangian step stable for any time step, even for a CFL number of 10? *Hint: Write the new value as a combination of old values and look at the weights. What can a convex combination do to the maximum?*
2. A whole-cell shift ($\alpha = 0$) produces no blur at all. Explain why a grid-aligned flow looks sharper than a diagonal one at the same speed. *Hint: Look at the fractional part of the shift in x and in y separately. A diagonal flow almost never has both equal to zero.*
3. Using $\nu_{\text{num}} = \alpha(1 - \alpha)h^2/(2\Delta t)$, how does the blur change when you halve the time step at fixed speed and grid? *Hint: The shift per step halves, so α changes, and so does the number of steps to reach the same time. Work out the variance gained over one fixed time.*

Evaluate

1. Compare semi-Lagrangian and MacCormack for a smoke simulation in which the smoke has sharp edges and for a simulation in which mass must be exactly conserved. Which would you choose for each, and why? *Hint: MacCormack with the clamp is sharper but not conservative: the mass run on this page rose by about one percent.*
2. A colleague doubles the grid resolution to reduce the blur but leaves the time step unchanged. Does the numerical diffusion in physical units fall by a factor 4? *Hint: Check how the fractional shift α changes when h halves at fixed speed and Δt .*

Create

1. Design a test that measures the numerical diffusion of the workspace's advection for the velocity field itself (not a dye). What would you advect and what would you measure?
2. Propose a limiter that is less crude than the four-cell clamp, for example one that checks the sign of the correction. What would you test it on?

Check yourself

1. understand: In one sentence each: why does the semi-Lagrangian step never blow up, and why does it blur?
Hint: Convex combination.
Answer: It never blows up because every new value is an average of old values with weights between 0 and 1, so no value can exceed the old maximum. It blurs because averaging two neighbouring cells is exactly what a diffusion does, and the fractional part of the shift decides how much is mixed.

2. apply: A Gaussian of width 3 cells is shifted half a cell per step for 100 steps. What width does linear interpolation leave, in cells?
Hint: Variance grows by $\alpha(1 - \alpha)$ per step.
Answer: $\sigma = \sqrt{9 + 100 \times 0.25} = 5.831$ cells; the Flow solver measured 5.831.
3. analyze: In the Flow solver, the same run with MacCormack and the clamp ends with width 3.53 cells and peak 0.83, while semi-Lagrangian ends with 5.83 and 0.514. Why is the MacCormack peak still below 1?
Hint: The clamp holds the result inside the range of the four cells it came from.
Answer: A peak is a local maximum, and the clamp limits the corrected value to the range of the surrounding old values, so near the peak the scheme falls back toward the first-order step. The correction cancels the blur away from the extremum but cannot restore the original height there.
4. evaluate: MacCormack with the clamp is sharper but its mass changed by about one percent in the test. Is that acceptable for a smoke hero? For a combustion simulation?
Hint: What does each simulation do with the dye or the scalar?
Answer: For a visual effect where the dye only makes the flow visible, a one percent drift is invisible and the sharper picture is worth it. For a combustion or species-transport simulation, mass of a species is a physical quantity: use a conservative scheme (a flux-form finite-volume scheme) even though it costs a time-step limit.
5. create: Design an experiment in the Flow workspace that shows numerical diffusion in the velocity, not just in the dye.
Hint: Start from a shear layer, switch off confinement and viscosity.
Answer: Use the Shear layer scene with confinement 0, viscosity 0 and velocity dissipation 0, and watch the vorticity view: the vorticity peak falls although nothing physical diffuses it. Repeat with semi-Lagrangian and with MacCormack, and at 32, 64 and 96 cells: the peak after a fixed time measures the artificial viscosity of each combination.

Numbers on this page and where they come from

Quantity	Value	Source
Semi-Lagrangian, half-cell shift, 100 steps: Gaussian width (initial 3 cells)	5.8307 cells	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js), $N = 96$
Same, theory $\sqrt{9 + 100 \times 0.25}$	5.831 cells	tools/flow_numbers.py (numpy / sympy)
Same, peak height (initial 1)	0.5138	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js)
MacCormack with the clamp, half-cell shift: width and peak	3.526 cells and 0.8302	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js)
Quarter-cell shift, semi-Lagrangian: width (theory $\sqrt{9 + 100 \times 0.1875} = 5.2678$)	5.2676 cells	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js)
Mass change after 100 steps: semi-Lagrangian, MacCormack	0.000 %, 1.00 %	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js) against the analytic sum

Quantity	Value	Source
Numerical diffusion, $\alpha = 1/2$, $h = 1/96$, $dt = 1/60$ s	0.0008138	tools/flow_numbers.py (numpy / sympy)
Check: $2 \nu t$ in cells squared after 100 steps	25	tools/flow_numbers.py (numpy / sympy) (equals 100 x 0.25)
Largest $ g $ over α in $[0,1]$ and θ in $[0,\pi]$	1	tools/flow_numbers.py (numpy / sympy)
CFL of the hero stream (0.36 per s) on 128 and on 64 cells, $dt = 1/60$ s	0.768 and 0.384	tools/flow_numbers.py (numpy / sympy)

Sources

- Courant, R.; Friedrichs, K.; Lewy, H. (1928). Über die partiellen Differenzengleichungen der mathematischen Physik. Mathematische Annalen.

2 Jacobi iteration smooths the pressure but converges at a rate $\cos(\pi/(N+1))$: 6 to 24 sweeps are nowhere near a solve

Jacobi replaces every pressure by the average of its neighbours corrected by the divergence. The error shrinks by the spectral radius of the iteration matrix each sweep, and for an N by N grid that is $\cos(\pi/(N+1))$: just below one, so thousands of sweeps are needed to solve and the 6 to 24 sweeps of the hero only smooth. Conjugate gradients, Gauss-Seidel with over-relaxation and multigrid change the rate; the residual plot of the Flow workspace shows the difference.

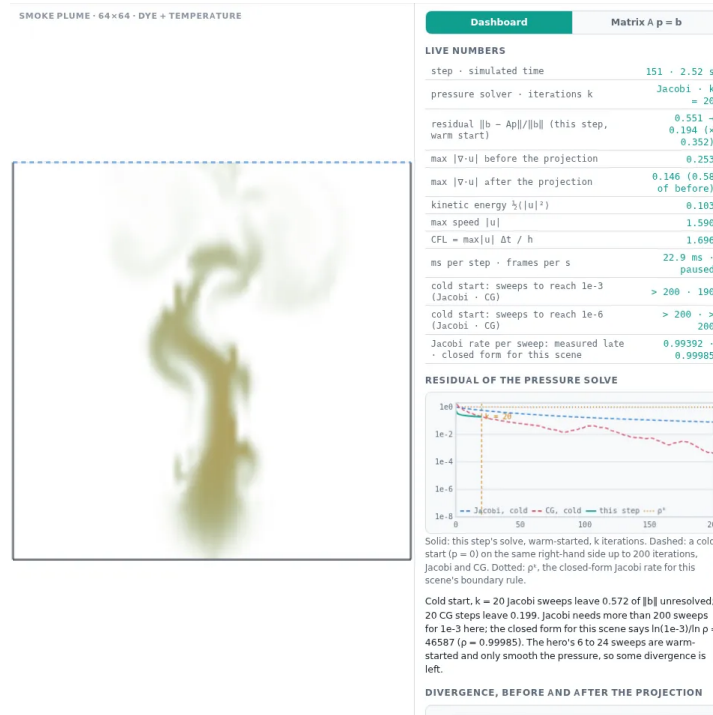


Figure 2: Screenshot of the Dashboard of the Flow workspace: the relative residual of the pressure solve against iteration, for the current step (solid) and for a cold start with Jacobi and with conjugate gradients (dashed).

Read the figure

Question. How fast does each iterative solver reduce the residual $\|b - Ap\| / \|b\|$, and how many iterations are enough?

Axes. x: iteration number k , from 0 to 200. y: the relative residual on a logarithmic scale from $1e-8$ to 2: each grid line is two decades. The dotted line is ρ^k with the closed-form Jacobi rate. The orange vertical line is the number of iterations the simulation uses per step.

Look at.

- The slope of the Jacobi curve: a straight line on a log plot means a constant factor per sweep, the spectral radius.
- Where the conjugate-gradient curve crosses the Jacobi curve: CG starts slower and then overtakes it.
- How much lower the solid curve starts than the dashed ones: the warm start $p < 0.8 p$ begins from last step's pressure.
- The orange line against the curves: the residual left at the number of iterations actually used.

Must check.

- Warm or cold start for each curve.
- The norm of the residual (the 2-norm over the fluid cells here) and the normalisation by $\|b\|$.
- The boundary rule: it changes the rate (open box $\cos(\pi/(N+1))$; the plume's walls plus open top much closer to 1).

If it is wrong.

- A plot on a linear axis hides everything below $1e-2$: the whole story is in the decades.
- Comparing iteration counts of Jacobi and CG as if they cost the same: one CG step costs about twice a Jacobi sweep.
- Starting one curve from a warm start and another from zero without saying so.

What it proves. How much of the pressure equation each solver has solved at a given cost. It does not prove that the flow is right: a small residual can still hide an error as large as the condition number times it.

Terms

Jacobi iteration $p_c^{(k+1)} = \frac{1}{d_c} \left(\sum_{c' \sim c} p_{c'}^{(k)} - b_c \right)$ Update every unknown from its neighbours' values of the previous sweep, all at once. Needs no ordering and parallelises perfectly, which is why shaders use it.

Residual $r = b - Ap$ How far the current guess is from solving the equations. The error is the distance from the exact solution, which the residual only bounds.

Iteration matrix $R = I - D^{-1}(-A)$ The matrix that turns the error of one sweep into the error of the next.

Spectral radius $\rho(R)$ The largest size of an eigenvalue of the iteration matrix. The error shrinks by this factor per sweep in the long run.

Smoother An iteration that damps the rough (high-frequency) part of the error quickly and the smooth part slowly.

Gauss-Seidel and SOR $\omega_{\text{opt}} = \frac{2}{1 + \sin \frac{\pi}{N+1}}$ Use the new values as soon as they exist (Gauss-Seidel); extrapolate each update by a factor omega (successive over-relaxation).

Conjugate gradients $\|e_k\|_A \leq 2 \left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1} \right)^k \|e_0\|_A$ A Krylov method for symmetric positive definite systems: each step minimises the error in the energy norm over all the search directions so far.

Warm start $p^{(0)} = 0.8 p^{n-1}$ Begin the iteration from the previous step's pressure (scaled by 0.8 in the hero) instead of zero, because pressure changes slowly in time.

Remember

- Jacobi on the open $N \times N$ grid: $\rho(R) = \cos \frac{\pi}{N+1}$. Error after k sweeps $\sim \rho^k$; sweeps to gain a factor ε : $\ln \varepsilon / \ln \rho$.
- $N = 6$: $\rho = 0.90097$, 66 sweeps for 10^{-3} . $N = 64$: $\rho = 0.99883$, 5912 sweeps. $N = 96$: $\rho = 0.99948$, 13168 sweeps.
- For small $1 - \rho$: $1 - \rho \approx \pi^2 / (2(N+1)^2)$, so the sweeps needed grow like N^2 and the work like N^4 .

- Gauss-Seidel: $\rho_{GS} = \rho_J^2$ for this matrix, half the sweeps. SOR with the optimal ω : $\rho = \omega - 1$. CG: about $\frac{1}{2}\sqrt{\kappa} \ln(2/\varepsilon)$ steps.
- Plain Jacobi does not damp the highest-frequency modes better than the smooth ones (its rate there is also ρ); weighted Jacobi with $\omega = 0.8$ does: smoothing factor 0.60.
- A small residual does not mean a small error: $\|e\|/\|x\| \leq \kappa \|r\|/\|b\|$.

Understand

Easy: pass the news along one neighbour at a time Jacobi is a rumour mill. Each cell tells its neighbours its pressure, and each cell sets its own pressure to the average of what it hears, corrected for any fluid being created there. News from the far side of a grid 64 cells wide needs about 64 sweeps just to arrive, and many more times that to settle, so a few dozen sweeps pass the news only locally. That is enough to smooth the pressure, not enough to solve it.

Medium: the rate is a cosine Each sweep multiplies the slowest-decaying part of the error by $\rho = \cos(\pi/(N+1))$. For a 6 by 6 grid that is 0.9010, and 66 sweeps give a factor 10^{-3} . For 64 by 64 it is 0.99883, and reaching 10^{-3} takes 5912 sweeps. The hero's 6 to 24 sweeps reduce the slowest error by only $\rho^{20} = 0.977$, a loss of about 2.3 % at $k = 20$.

Hard: how the solvers compare, with real numbers On the 64 by 64 open grid: Jacobi 5912 sweeps for 10^{-3} ; Gauss-Seidel 2956; SOR with $\omega = 1.9078$ about 71; CG about 157 steps. On the plume's own matrix (walls with an open top, rate 0.999852 in closed form and 0.999849 measured on a cold start) Jacobi had only reached 0.116 after 200 sweeps and CG 0.00415. Early on the order is reversed: after 24 iterations from zero Jacobi was at 0.252 and CG at 0.772, because CG minimises the energy norm of the error and its residual norm is not monotone.

Hardest: why Jacobi is not even a good smoother, and what a small residual hides The modes of the 5-point Jacobi matrix with the highest frequency in both directions have eigenvalue $-\rho$, so plain Jacobi does not damp the checkerboard better than the smoothest mode: its smoothing factor over the upper half of the frequencies is 0.99883. Weighting the update by $\omega = 0.8$ lowers it to 0.599, the reason multigrid uses weighted Jacobi or Gauss-Seidel as its smoother. And a residual of 10^{-3} does not guarantee a relative error of 10^{-3} : the bound is $\kappa \times 10^{-3} = 1.71$ for $N = 64$. Judging a pressure solve by its residual alone can be misleading when the matrix is ill conditioned.

Derivation

Step 1. Split the matrix

$$A = D + (A - D), \quad D = \text{diag } A = -\text{diag}(d_c)$$

D is -4 in the interior. Move everything off the diagonal to the right-hand side.

Step 2. The Jacobi update

$$D p^{(k+1)} = b - (A - D) p^{(k)} \implies p^{(k+1)} = p^{(k)} + D^{-1}(b - A p^{(k)})$$

In the grid form this is the average of the four neighbours minus $b/4$ (the formula in the hero's pressure shader).

Step 3. The error obeys a linear recursion

$$e^{(k+1)} = R e^{(k)}, \quad R = I - D^{-1}(-A) \implies e^{(k)} = R^k e^{(0)}, \quad \|e^{(k)}\| \lesssim \rho(R)^k \|e^{(0)}\|$$

Expand $e^{(0)}$ in eigenvectors of R : each is multiplied by its eigenvalue to the power k . The slowest one dominates.

Step 4. The eigenvectors are sine waves

$$v_{k\ell}(i, j) = \sin \frac{k\pi i}{N+1} \sin \frac{\ell\pi j}{N+1}, \quad \lambda_{k\ell} = \frac{1}{2} \left(\cos \frac{k\pi}{N+1} + \cos \frac{\ell\pi}{N+1} \right)$$

Insert into the stencil with $p = 0$ at the boundary and use $\sin(a+b) + \sin(a-b) = 2 \sin a \cos b$.

Step 5. The spectral radius

$$\rho(R) = \lambda_{11} = \cos \frac{\pi}{N+1} \approx 1 - \frac{\pi^2}{2(N+1)^2}, \quad k_\varepsilon = \frac{\ln \varepsilon}{\ln \rho}$$

The Matrix tab checks it two ways: the eigenvalues of the actual matrix agree with the cosine to 1e-13, and the measured convergence rate of a real Jacobi run agrees with it to six digits.

Step 6. Conjugate gradients, for comparison

$$\|e_k\|_A \leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k \|e_0\|_A, \quad k_\varepsilon \approx \frac{1}{2} \sqrt{\kappa} \ln \frac{2}{\varepsilon}$$

With $\kappa = 1712$ for $N = 64$ that is about 157 steps for 10^{-3} , against 5912 for Jacobi.

Worked example: How many Jacobi sweeps to reduce the error by 1000 on the home grid?

The hero's pressure grid is 128 by 128 in its second tier and 64 by 64 in its smallest. For the 64 by 64 open grid, compute the Jacobi spectral radius, the number of sweeps for a factor 1000 and the reduction after 20 sweeps. Compare with CG.

1. Spectral radius.

$$\rho = \cos \frac{\pi}{65} = 0.998832$$

2. Sweeps for a factor 10^{-3} .

$$k = \frac{\ln 10^{-3}}{\ln \rho} = 5912$$

3. Error left after 20 sweeps in the slowest mode.

$$\rho^{20} = 0.9769$$

4. CG: the condition number is $\kappa = \cot^2(\pi/130) = 1712$, so about

$$k \approx \frac{1}{2} \sqrt{\kappa} \ln 2000 = 157$$

Answer. $\rho = 0.998832$; about 5912 sweeps for a factor 1000; 20 sweeps leave 0.977 of the slowest error; CG needs about 157 steps. The code below checks the first two against the actual matrix.

Analyze

1. Why does the number of Jacobi sweeps needed grow like N^2 while the cost of one sweep grows like N^2 as well? What is the total work as a function of N ? *Hint: $1 - \rho \approx \pi^2 / (2(N + 1)^2)$, so the sweeps for a fixed reduction scale as N^2 ; one sweep touches N^2 cells.*
2. Jacobi is perfectly parallel and Gauss-Seidel is twice as fast per sweep. Why do the hero's shaders use Jacobi? *Hint: A fragment shader writes every pixel of one texture from the pixels of another. What does Gauss-Seidel need that a shader cannot give it?*
3. The residual after 24 iterations of CG was 0.77 and of Jacobi 0.25 on the same cold start, but after 200 iterations CG was ahead by a factor 28. Explain. *Hint: CG minimises the A-norm of the error over a growing Krylov space. The 2-norm of the residual is not monotone.*

Evaluate

1. The hero spends 6 to 24 sweeps per frame. Is it reasonable to call its pressure 'solved'? What does it deliver instead? *Hint: With the plume field the divergence after the projection was 0.55, 0.51 and 0.51 of before for 6, 24 and 80 sweeps: the results hardly differ.*
2. You want a pressure solve for a 512 by 512 simulation that is exact to $1e-6$ every frame. Which method would you pick and why? *Hint: Compare Jacobi, CG with a preconditioner, and multigrid for the cost per frame and for how well each maps to a GPU.*

Create

1. Add a weighted Jacobi option ($\omega = 0.8$) to the Flow workspace and predict how the residual curve changes at low iteration counts before you try it.
2. Design a measurement of the effective iteration count the hero needs so that the divergence after the projection stops improving. What would you plot against what?

Check yourself

1. understand: Why does Jacobi converge slowly on a large grid, in words?
Hint: How far does information travel in one sweep?
Answer: A cell hears only from its four neighbours each sweep, so information crosses the grid at one cell per sweep. The smooth, long-wavelength part of the error needs to spread over the whole grid and decays by only a factor just below one per sweep, $\cos(\pi/(N+1))$, so thousands of sweeps are needed on a grid that is tens of cells wide.
2. apply: What is the Jacobi spectral radius for the 64 by 64 open grid, and how many sweeps reduce the error by 1000?
Hint: $\cos(\pi/65)$; $\ln(1e-3)/\ln(\rho)$.
Answer: $\rho = 0.998832$; $k = 5912$ sweeps.
3. analyze: In the Flow workspace the plume scene's cold-start Jacobi rate was measured at 0.999849 per sweep, not 0.998832. Why?
Hint: The plume box has walls on three sides and an open top: a different boundary rule from the closed Dirichlet box.
Answer: With walls on three sides the slowest mode needs only a quarter wave across the box instead of a half wave, and the closed form becomes $(1 + \cos(\pi/(2N+1)))/2 = 0.999852$ for $N = 64$, which matches the measured 0.999849 to $3e-6$. Fewer open sides means a slower solve: the home-page geometry is about 8 times slower than the all-open box.

4. evaluate: After 200 sweeps from zero the Flow solver's relative residual on the plume field was 0.12 for Jacobi and 0.0042 for CG. A reviewer says: this shows Jacobi is a bad solver. Is that the right conclusion for the hero?

Hint: The hero solves a warm-started problem for a few iterations a frame and only needs a smooth pressure.

Answer: Not by itself. For a full solve to a small residual CG is far better (by a factor of about 28 here at 200 iterations, and the gap widens). But the hero needs only a smooth pressure that changes slowly from frame to frame, a few Jacobi sweeps from a warm start do that, each sweep is one cheap shader pass with no global dot products (which CG needs and a GPU does badly), and the divergence after the projection hardly improves with more iterations because of the stencil floor.

5. create: Design a fair benchmark of Jacobi against CG for this workspace that accounts for the cost of one iteration.

Hint: Count operations, not iterations.

Answer: Count matrix-vector products and dot products: a Jacobi sweep is one stencil pass, a CG step is one stencil pass plus two dot products and three vector updates, about twice the work of a sweep. Plot the residual against work in units of one stencil pass, from a cold start and from the warm start, for several grid sizes and both boundary rules, and report the work to reach $1e-3$ and $1e-6$. The curves of this page, redrawn with CG's x axis doubled, are the first version.

Numbers on this page and where they come from

Quantity	Value	Source
Jacobi spectral radius, open $N \times N$ grid: $N = 6, 32, 64, 96$	0.900969, 0.995472, 0.998832, 0.999476	tools/flow_numbers.py (numpy / sympy) ($\cos(\pi/(N+1))$)
Sweeps for an error factor $1e-3$: $N = 6, 32, 64, 96$	66, 1522, 5912, 13168	tools/flow_numbers.py (numpy / sympy)
Error left after 20 sweeps in the slowest mode, $N = 64$	0.9769	tools/flow_numbers.py (numpy / sympy)
rho of the actual matrix against $\cos(\pi/(N+1))$, $N = 3$ to 12	agree to $4.8e-14$	tests/js/flow_operator_test.mjs (symmetric eigenvalue solve of the dense matrix)
Gauss-Seidel: rho and sweeps for $1e-3$, $N = 64$	0.997666, 2956	tools/flow_numbers.py (numpy / sympy)
SOR: optimal omega, rho and sweeps for $1e-3$, $N = 64$	1.9078, 0.9078, 71	tools/flow_numbers.py (numpy / sympy)
Condition number and CG steps for $1e-3$, $N = 64$	1712, 157	tools/flow_numbers.py (numpy / sympy) ($\cot^2(\pi/130)$, 0.5 $\sqrt{\kappa} \ln 2000$)
Smoothing factor over the high frequencies, $N = 64$: plain Jacobi, weighted omega = 0.8	0.99883, 0.5991	tools/flow_numbers.py (numpy / sympy)
Bound on the relative error from a residual of $1e-3$, $N = 64$	1.71	tools/flow_numbers.py (numpy / sympy) ($\kappa \times 1e-3$)
Plume scene (walls + open top), $N = 64$: closed-form rho and measured late rate	0.9998517 and 0.9998494	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js) (12000 cold-start sweeps)

Quantity	Value	Source
Plume field, cold start, residual after 24 and after 200 iterations: Jacobi	0.252 and 0.116	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js)
Same, conjugate gradients	0.772 and 0.00415	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js)

Sources

- Trefethen, L. N.; Bau, D. (1997). Numerical Linear Algebra. SIAM.
- Press, W. H.; Teukolsky, S. A.; Vetterling, W. T.; Flannery, B. P. (2007). Numerical Recipes: The Art of Scientific Computing. Cambridge University Press.

3 Incompressible flow: the velocity must have zero divergence, and the pressure is whatever enforces it

Water and slow air do not compress, so the velocity field has zero divergence at every instant. Pressure has no equation of its own: it is the field whose gradient, subtracted from the velocity, removes whatever divergence the other terms create. Splitting a time step into 'move things around' and 'project onto the divergence-free fields' is the idea behind the home-page hero and behind the Flow workspace.

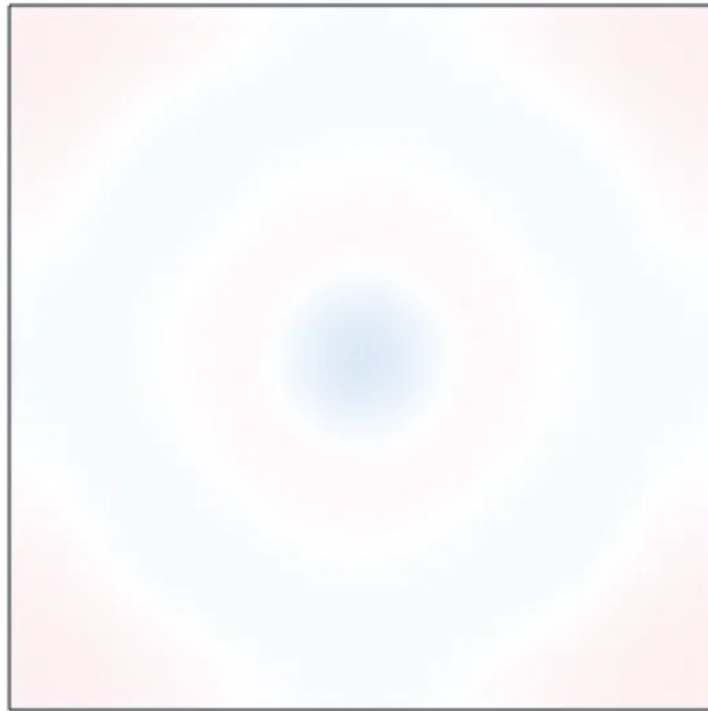


Figure 3: Screenshot of the Flow workspace (scene 4, divergence view). A spreading source and a swirl were put in; the projection has removed the source, so what is left of the divergence is small.

Read the figure

Question. After a pressure projection, how much of the velocity's divergence is left, and where?

Axes. The square is the simulation box (x to the right, y up). Colour is the divergence of the velocity after the projection: rose negative (a sink), blue positive (a source), dark near zero. The scale is set by the largest value on screen, so the colour alone does not say how large the divergence is.

Look at.

- Whether the colours are concentrated or spread evenly: a left-over divergence that follows the cell grid is a discretisation effect, not a physical source.
- The Dashboard numbers beside the picture: max divergence before and after the projection.
- Whether the swirl is still there when the Show menu is switched to vorticity: the projection must not touch it.

Must check.

- Which stencil the divergence was computed with (here central differences over two cells).

- The number of pressure iterations and the solver (Jacobi or conjugate gradients).
- The boundary condition on every side.

If it is wrong.

- A colour scale that is silently re-normalised each frame makes a tiny divergence look as dramatic as a large one: read the numbers, not only the colours.
- Looking at the divergence of the velocity before the projection and calling it the result: only the after-field matters.

What it proves. That the incompressibility constraint was imposed to the accuracy the pressure solver reached, not that the flow is physically right.

Terms

Velocity field $u(x, t) = (u, v)$ The velocity of the fluid at every point and time. On the grid it is stored at the cell centres, in domain lengths per second.

Pressure $p(x, t)$ A scalar field whose gradient pushes the fluid. In incompressible flow it adjusts instantly so that the velocity stays divergence free.

Divergence $\nabla \cdot u = \partial_x u + \partial_y v$ The net outflow per unit volume at a point: positive where fluid spreads out, negative where it gathers. Zero everywhere means no volume is created or destroyed.

Incompressible $\nabla \cdot u = 0$ The volume of a fluid parcel does not change as it moves, so the velocity field is divergence free.

Advection term $(u \cdot \nabla)u$ The part of the momentum equation that carries the velocity along with itself. It is the only nonlinear term.

Kinematic viscosity ν How fast momentum diffuses, in length squared per time. Air at 20 degrees C has about $1.5 \times 10^{-5} \text{ m}^2/\text{s}$, water about 1.0×10^{-6} .

Reynolds number $\text{Re} = \frac{UL}{\nu}$ The ratio of inertial to viscous effects for a flow of speed U over a length L . Large values mean thin boundary layers and eddies.

Helmholtz-Hodge decomposition $u^* = w + \nabla \phi$, $\nabla \cdot w = 0$ Any smooth vector field splits into a divergence-free part plus the gradient of a scalar. The two parts are orthogonal when the boundary condition is right.

Projection $\mathbb{P} u^* = w$ The operator that keeps the divergence-free part and throws the gradient part away. Pressure is how it is computed.

Boundary condition $u \cdot n = 0$, $p = 0$ What the fluid does at the edge: a wall allows no flow through it (slip: sliding along it is free; no-slip: velocity zero), an open side lets the flow out with the pressure fixed.

Remember

- Momentum: $\partial_t u + (u \cdot \nabla)u = -\frac{1}{\rho} \nabla p + \nu \nabla^2 u + f$. Constraint: $\nabla \cdot u = 0$.
- In two dimensions there are three unknowns (u, v, p) and three equations (two momentum components and the constraint). On a 64 by 64 grid that is 12,288 unknowns.
- Pressure has no time derivative and no equation of its own: it is a Lagrange multiplier for the constraint.
- Every field splits as $u^* = w + \nabla \phi$ with $\nabla \cdot w = 0$; the projection returns w .

- A stream function gives a divergence-free field for free: $u = \partial_y \psi$, $v = -\partial_x \psi \Rightarrow \nabla \cdot u = 0$. A swirl is untouched by the projection; a pure expansion is removed completely.
- Reynolds number $Re = UL/\nu$: about 6596 for 1 m/s of air over 0.1 m, about 99602 for water, 250 for the lid-driven cavity scene.

Understand

Easy: a corridor where nobody can be squeezed Picture a crowded corridor. If the people at one end push forward, everyone ahead has to move at once, and if a gap opens somewhere someone has to fill it. Incompressible fluid is that corridor: volume cannot be created or destroyed, so the velocity field can have no source and no sink. Pressure is the invisible hand that passes a push along instantly.

Medium: three unknowns, three equations Divergence measures the net outflow per volume. A swirl has none: what leaves one side comes back on the other. An expanding blob has positive divergence. The two momentum equations tell how the velocity changes, but they know nothing about the constraint, so after a step the field is no longer divergence free. The third equation, $\nabla \cdot u = 0$, fixes the pressure: it must be exactly the field that cancels the divergence the step created.

Hard: the projection keeps the swirl and removes the push Write the predicted velocity as a divergence-free part plus a gradient, $u^* = w + \nabla \phi$. Subtracting $\nabla \phi$ leaves w , and finding ϕ means solving a Poisson equation with the divergence of u^* on the right. In the Projection test scene a spreading source and a swirl were added together. With an exact solve the projection removed 59.0 % of the kinetic energy (0.001533 to 0.0006284) and left the swirl alone.

Hardest: divergence free on a grid is a different statement On a grid the divergence is a finite difference. With central differences, $\nabla \cdot (\nabla^\perp \psi)$ is zero to rounding (2.8e-17 in the solver), so swirls are exactly invisible to the projection. But the pressure is found from the compact five-point Laplacian, whereas the divergence of the gradient, with the same central differences, is a wider stencil (spacing two cells). They agree for smooth pressure and disagree for rough pressure, so even an exact solve leaves a floor: 0.0024 of the original divergence for a smooth field, about half for the rough plume field. A staggered grid removes the mismatch; the hero keeps the collocated grid for speed and simplicity.

Derivation

Step 1. Mass is conserved

$$\frac{d}{dt} \int_V \rho dV = - \oint_{\partial V} \rho u \cdot n dS \Rightarrow \partial_t \rho + \nabla \cdot (\rho u) = 0$$

For constant density this is $\nabla \cdot u = 0$: the net flux through any closed surface is zero.

Step 2. Momentum balance for a fluid parcel

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

Newton's second law per unit mass: the pressure gradient, the viscous diffusion of momentum and a body force f (gravity, buoyancy) accelerate the parcel.

Step 3. Take the divergence of the momentum equation

$$\nabla^2 p = -\rho \nabla \cdot ((u \cdot \nabla)u) + \rho \nabla \cdot f$$

Using $\nabla \cdot u = 0$ at all times: the time derivative and the viscous term drop out (the divergence commutes with the Laplacian). Pressure obeys a Poisson equation whose source is made of the other terms.

Step 4. Split the time step (Chorin)

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

First move the fluid as if it could be compressed, then project. The Flow workspace does the first part with Stam's semi-Lagrangian advection and the forces of the hero.

Step 5. Why the projection is a projection

$$\nabla \cdot u^{n+1} = \nabla \cdot u^* - \frac{\Delta t}{\rho} \nabla^2 p = 0, \quad \int_V w \cdot \nabla \phi \, dV = - \int_V \phi \nabla \cdot w \, dV + \oint_{\partial V} \phi w \cdot n \, dS = 0$$

The second identity says that divergence-free fields tangent to the walls are orthogonal to gradients, so what is subtracted is exactly the gradient part and nothing else.

Step 6. On the grid: one sparse linear system

$$A p = b, \quad b_c = \frac{1}{2} (u_R - u_L + v_T - v_B)_c, \quad u_c^{n+1} = u_c^* - \frac{1}{2} (p_R - p_L, p_T - p_B)_c$$

The same three passes as the hero's divergence, pressure and gradient shaders. Open the scene to see the divergence before and after.

Worked example: Project a field by hand, then check it with an FFT

In the periodic box $[0, 2\pi)^2$ the predicted velocity is $u^* = (\sin x + \sin y, 0)$. Find the divergence-free velocity after the projection and the kinetic energy it keeps.

1. Divergence of u^* : only $\partial_x u$ contributes, since $v = 0$.

$$\nabla \cdot u^* = \cos x$$

2. Solve $\nabla^2 \phi = \cos x$. A function of x alone with $\phi'' = \cos x$ is

$$\phi = -\cos x$$

3. Its gradient is the part to remove.

$$\nabla \phi = (\sin x, 0)$$

4. Subtract it.

$$u = u^* - \nabla \phi = (\sin y, 0)$$

5. Check: u depends only on y and has no y component, so its divergence is zero. The kinetic energy per unit area falls from $\frac{1}{2}(\frac{1}{2} + \frac{1}{2})$ to $\frac{1}{2} \cdot \frac{1}{2}$.

$$\frac{1}{2} \langle |u^*|^2 \rangle = 0.5 \longrightarrow \frac{1}{2} \langle |u|^2 \rangle = 0.25$$

Answer. The projected velocity is $(\sin y, 0)$, a shear flow, and the kinetic energy halves (0.5 to 0.25). The numpy FFT code below gives the same: error 5.6e-16, energies 0.5 and 0.25, divergence after 5e-15.

Analyze

1. Why is the swirl $u = (-y, x)$ unchanged by the projection while $u = (x, y)$ is removed completely? *Hint: Compute the divergence and the curl of each. The first has zero divergence; the second is the gradient of $(x^2 + y^2)/2$.*
2. Show that a gradient field $\nabla\phi$ carries no circulation, so the projection can never remove a vortex. *Hint: The circulation $\oint \nabla\phi \cdot d\mathbf{l}$ around a closed loop is the net change of ϕ , which is zero.*
3. Two equal jets collide head on. What does the Poisson equation say about the pressure at the stagnation point, and why does the fluid turn sideways? *Hint: The right-hand side contains $-\rho\nabla\cdot((u\cdot\nabla)u)$. Near a stagnation point $u \approx (-ax, ay)$; evaluate it. Pressure rises to slow the streams and its gradient pushes them out sideways.*

Evaluate

1. A friend says incompressible means the density is the same everywhere. Is that right? *Hint: Think of a smoke plume. The density varies in space (hot gas is lighter) but each parcel keeps its density as it moves, $D\rho/Dt = 0$. In the Boussinesq model the variation appears only in the buoyancy force.*
2. At what speed does the assumption $\nabla\cdot u = 0$ stop being a good one for air? *Hint: The density change along a streamline is of order $Ma^2/2$. At Mach 0.3 that is 4.5 %, the usual rule of thumb for the limit.*

Create

1. Design a scene for the Flow workspace in which a missing projection is obvious at a glance (which fields, which view, which preset values?), and a second in which an under-converged projection is obvious but a missing one is not.
2. Replace the closed box of the projection scene by one with an open side. Predict what happens to the pressure and to the divergence at that side before you try it.

Check yourself

1. understand: Why does the pressure in an incompressible flow have no time derivative and no equation of its own?
Hint: Count unknowns and equations, and ask what the third equation constrains.
Answer: In two dimensions there are three unknowns (u, v, p) and three equations: two momentum components and the constraint $\text{div } u = 0$. The constraint contains no pressure, so pressure is found as the field that makes the other two keep the constraint true: it is a Lagrange multiplier, and it acts instantly because the constraint must hold at every moment.
2. apply: The lid-driven cavity scene has a lid speed of 0.5 domain lengths per second, a box of side 1 and viscosity 0.002. What is its Reynolds number?
Hint: $Re = U L / \nu$.
Answer: $Re = 0.5 \times 1 / 0.002 = 250$.
3. analyze: A predicted velocity is $u^* = (\sin x + \sin y, 0)$ in a periodic box. Which part does the projection remove, and what fraction of the kinetic energy goes with it?
Hint: Compute the divergence. Then ask which of the two terms depends on the direction it points along.
Answer: The $\sin x$ term in the x velocity (divergence $\cos x$) is a pure gradient and is removed; the $\sin y$ term is a shear with no divergence and stays. Each term carries half of the energy, so the energy falls from 0.5 to 0.25, which an FFT projection reproduces to 5.6e-16.

4. evaluate: In the Projection test scene, an exact solve cuts the maximum divergence from 0.0732 to 0.00131 but not to zero. Is the projection wrong?

Hint: Compare the compact five-point Laplacian that A is built from with the divergence of the central-difference gradient.

Answer: No. It removed 98.2 % of the divergence. The pressure is found from the compact five-point Laplacian but subtracted with central differences, whose divergence is a wider stencil, so a floor of rough-field divergence remains even for an exact solve (0.0024 of the original for a smooth field, about 0.5 for the plume field). A staggered grid has no such floor.

5. create: Describe an experiment on the Flow workspace that separates the error from too few pressure iterations from the error from the stencil mismatch.

Hint: One of them changes when the iteration count changes; the other does not.

Answer: Run the Projection test scene and vary the pressure iterations from 1 to 80, then switch to conjugate gradients with the same counts. The divergence after the projection keeps falling with the iteration count while the solve is unconverged (the residual plot shows this) and then stops at a floor that no iteration count lowers: that floor is the stencil mismatch. Repeating on a smooth initial field lowers the floor, which shows it depends on the roughness of the pressure.

Numbers on this page and where they come from

Quantity	Value	Source
Unknowns on a 64 by 64 grid	12,288	3 x 64 x 64
Reynolds number, air 1 m/s over 0.1 m ($\nu = 1.516\text{e-}5 \text{ m}^2/\text{s}$)	6596	tools/flow_numbers.py (numpy / sympy)
Reynolds number, water 1 m/s over 0.1 m ($\nu = 1.004\text{e-}6 \text{ m}^2/\text{s}$)	99602	tools/flow_numbers.py (numpy / sympy)
Reynolds number, cavity scene ($U = 0.5$, $L = 1$, $\nu = 0.002$)	250	tools/flow_numbers.py (numpy / sympy)
Spectral projection of $(\sin x + \sin y, 0)$: error against $(\sin y, 0)$	5.61e-16	numpy FFT, 64 x 64 periodic (the code on this page)
Kinetic energy before and after that projection	0.5 and 0.25	numpy FFT
Projection test scene: kinetic energy before and after an exact solve	0.001533 and 0.0006284	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js), 600 CG iterations, $N = 64$
Projection test scene: max divergence before and after	0.0732 and 0.00131	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js)
max $ \text{div}(\text{curl } \psi) $ with the solver's divergence	2.78e-17	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js), also tests/js/flow_operator_test.mjs
Divergence left after an exact solve, smooth field ($N = 32$)	0.00241 of before	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js)
Density change at Mach 0.3, $\text{Ma}^2/2$	4.5 %	arithmetic

Sources

- White, F. M. (2011). Fluid Mechanics. McGraw-Hill.

4 The projection solves one sparse matrix every step: minus four on the diagonal, plus one for each neighbour

Finding the pressure that removes the divergence is a linear system $A p = b$. A is the five-point Laplacian: a row for each cell with minus the number of neighbours on the diagonal and plus one for each neighbour, so only five of n entries in a row are nonzero. Walls, open sides and obstacles change the rows at the edge, and a box with only walls makes A singular. The Matrix tab of the Flow workspace draws the real matrix.

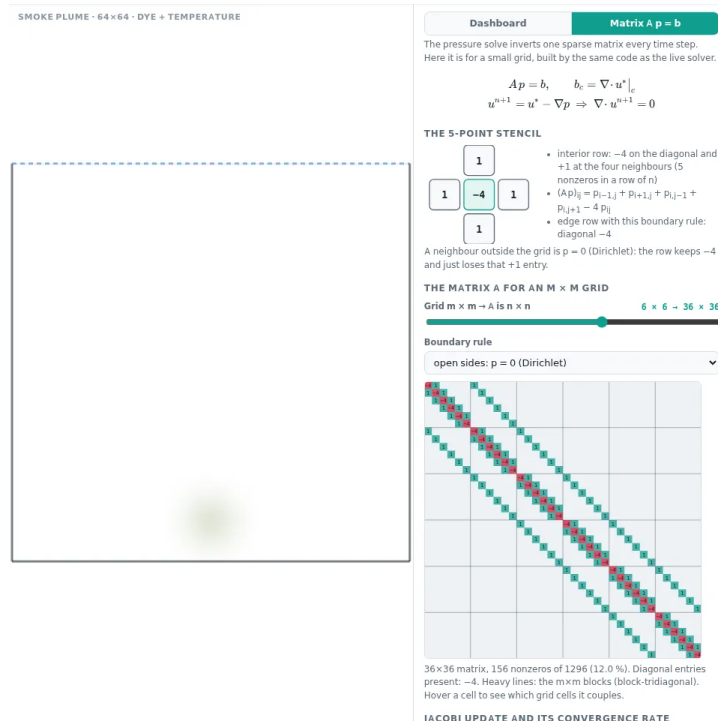


Figure 4: Screenshot of the Matrix tab of the Flow workspace: the 36 by 36 matrix of the pressure equation on a 6 by 6 grid with open sides, with its 156 nonzeros.

Read the figure

Question. Which cells does each equation of the pressure system couple, and what does the boundary change?

Axes. Row index down, column index across, both running through the cells of the grid line by line. A coloured square at (row, column) means that the equation of the cell in the row contains the pressure of the cell in the column. Rose: negative entry (the diagonal), teal: +1, dark: zero.

Look at.

- The diagonal: a constant minus four away from the boundary.
- The two bands next to the diagonal (left and right neighbour) and the two bands one block away (neighbours above and below): the block tridiagonal structure.
- The rows at a wall: with the wall rule the diagonal drops to minus three on an edge and minus two in a corner.
- How empty the matrix is: 156 nonzeros of 1296 entries for a 6 by 6 grid, 0.12 percent for the 64 by 64 grid of the home page.

Must check.

- Grid size m by m and the number of unknowns $n = m^2$.
- The boundary rule on each side.
- Row and column ordering (here: line by line).

If it is wrong.

- Reading the pattern as a dense matrix because it is drawn square: the cost of a solve depends on the nonzeros, not on n squared.
- Forgetting that the rows at the boundary depend on the boundary rule; a heat map without the rule stated is ambiguous.

What it proves. That every pressure value depends directly on its four neighbours only, so the matrix is sparse and a solve can be iterative.

Terms

Poisson equation $\nabla^2 p = \nabla \cdot u^*$ The equation for a field whose Laplacian is given. For pressure the given quantity is the divergence of the predicted velocity.

Five-point stencil $p_{i-1,j} + p_{i+1,j} + p_{i,j-1} + p_{i,j+1} - 4p_{i,j}$ The finite-difference Laplacian from the cell and its four neighbours: sum of the neighbours minus four times the cell.

Sparse matrix $\text{nnz} \approx 5n$ A matrix with almost all entries zero. It is stored and used as a list of neighbours, never as an n by n array.

Dirichlet condition $p = 0$ The value is fixed at the boundary. An open side fixes the pressure to zero, so the missing neighbour counts as 0 and the diagonal keeps its minus four.

Neumann condition $\partial_n p = 0$ The normal derivative is fixed. A wall means no pressure gradient across it: the missing neighbour is a copy of the cell, which folds its +1 into the diagonal.

Singular matrix $A\mathbf{1} = 0$ A matrix without an inverse. With walls on all sides adding a constant to the pressure changes nothing, so A times the constant vector is zero.

Compatibility condition $\sum_c b_c = 0$ For a singular system a solution exists only if the right-hand side has zero sum: what flows in must flow out. The workspace subtracts the mean of b before solving.

Condition number $\kappa(A)$ The ratio of the largest to the smallest eigenvalue size. It governs how slowly iterative solvers converge and how much a small residual can hide a large error.

Collocated grid A grid on which velocity and pressure live at the same points (the hero's choice). It is simple but the divergence and the Laplacian do not fit together exactly.

Remember

- $(Ap)_{i,j} = p_{i-1,j} + p_{i+1,j} + p_{i,j-1} + p_{i,j+1} - 4p_{i,j}$: diagonal -4 , four entries $+1$.
- Open $m \times m$ grid: $n = m^2$ unknowns and $5m^2 - 4m$ nonzeros: 156 for $m = 6$ ($n = 36$), 20,224 for $m = 64$ ($n = 4096$).
- Open side: missing neighbour is 0, diagonal stays -4 . Wall side: missing neighbour equals the cell, diagonal becomes -3 at an edge and -2 in a corner. Periodic side: the neighbour is on the other side.
- All walls: $A\mathbf{1} = 0$ (singular) and b must sum to zero.

- Condition number of the open $m \times m$ matrix: $\kappa = \cot^2 \frac{\pi}{2(m+1)}$: 19.20 for $m = 6$, 1712 for $m = 64$.
- The projection solves $Ap = b$ with b the central-difference divergence, then subtracts the central-difference gradient of p .

Understand

Easy: every cell asks its neighbours Pressure at a cell is set by one rule: it is the average of its four neighbours, corrected by how much fluid is trying to leave. Writing that rule for every cell at once gives a list of equations that each mention only a cell and its four neighbours. As a table of coefficients it is almost all zeros, a thin diagonal band, which is why the system can be solved without ever writing the table down.

Medium: the matrix, row by row In the interior a row has -4 on the diagonal and $+1$ in the four columns of the neighbours. Number the cells line by line and the left and right neighbours sit next to the diagonal, while the cells above and below sit a whole line (m columns) away: that is the block tridiagonal pattern you see in the Matrix tab. At the edge the rule decides: an open side counts the missing neighbour as zero, a wall copies the cell.

Hard: walls make the matrix singular With walls on every side, each row sums to zero (the diagonal is minus the number of neighbours), so the constant vector is in the null space: pressure is only defined up to an additive constant, and a solution exists only if the divergence sums to zero. The workspace subtracts the mean of b before solving. For the open grid the matrix is invertible with condition number $\cot^2(\pi/(2(m+1)))$: 19.2 at $m = 6$, 1712 at $m = 64$. The condition number grows like m^2 , which is why large grids are slow for simple iterations.

Hardest: the solve and the subtraction use different stencils The divergence and the gradient use central differences over two cells, so the divergence of the gradient is a Laplacian with spacing two, while A is the compact Laplacian with spacing one. The wide one cannot see a checkerboard pressure at all: applied to $(-1)^{i+j}$ it gives 0, while the compact one gives $-8p$ (size 8). So a checkerboard in the pressure is solved for but then invisible to the velocity update, and a rough right-hand side leaves a floor of divergence that more iterations cannot remove: for the plume field the ratio after to before was 0.546, 0.508, 0.507 for 6, 24 and 80 Jacobi sweeps and 0.519 for an exact solve. For a smooth field an exact solve leaves 0.0024.

Derivation

Step 1. Discretise the Laplacian on a uniform grid

$$\nabla^2 p \Big|_{i,j} \approx \frac{p_{i-1,j} + p_{i+1,j} + p_{i,j-1} + p_{i,j+1} - 4p_{i,j}}{h^2}$$

Second-order accurate: Taylor expand $p_{i\pm 1,j}$ and add. The factor $1/h^2$ is absorbed into the scaling of p in the workspace.

Step 2. Write one equation per fluid cell

$$(Ap)_c = \sum_{c' \sim c} p_{c'} - d_c p_c = b_c, \quad d_c = \# \text{fluid neighbours} + \# \text{open neighbours}$$

A wall or a solid neighbour contributes nothing (its pressure equals p_c), an open neighbour contributes $-p_c$ to the diagonal only.

Step 3. The matrix of an m by m open grid

$$A = \begin{pmatrix} T & I & & \\ I & T & I & \\ & I & T & \ddots \\ & & \ddots & \ddots \end{pmatrix}, \quad T = \begin{pmatrix} -4 & 1 & & \\ 1 & -4 & 1 & \\ & 1 & -4 & \ddots \\ & & \ddots & \ddots \end{pmatrix}$$

m by m blocks. Counting entries: m^2 on the diagonal and $4m(m-1)$ off it, so $5m^2 - 4m$ in all.

Step 4. Change the boundary rule, change the rows

$$d_c = \begin{cases} 4 & \text{interior, or an open side} \\ 3 & \text{wall edge} \\ 2 & \text{wall corner} \end{cases} \quad A\mathbf{1} = 0 \text{ for walls only}$$

Add up a wall row: the entries sum to zero, so the all-ones vector solves $Ax = 0$.

Step 5. Right-hand side and compatibility

$$b_c = \frac{1}{2}(u_R - u_L + v_T - v_B)_c, \quad \sum_c b_c = 0 \text{ required when } A\mathbf{1} = 0$$

For a singular system the workspace subtracts the mean of b , the discrete form of 'what flows in must flow out'.

Step 6. Correct the velocity

$$u_c^{n+1} = u_c^* - \frac{1}{2}(p_R - p_L, p_T - p_B)_c$$

Central-difference gradient. Its divergence is the wide Laplacian, which is why the discrete divergence does not become exactly zero.

Worked example: Build the matrix of a 3 by 3 open grid and apply it

Write the 9 by 9 matrix A for a 3 by 3 grid with open sides, numbering the cells line by line (cell 0 is bottom left). Apply it to $p = (1, 0, 0, 0, 0, 0, 0, 0, 0)$, a unit pressure in the bottom-left corner cell.

1. Every diagonal entry is -4 (open sides: the missing neighbour is zero but the diagonal keeps its -4). The off-diagonal $+1$ entries mark the neighbours: cell 0 touches cells 1 and 3, cell 4 touches 1, 3, 5, 7.

$$A_{00} = -4, A_{01} = A_{03} = 1, A_{44} = -4, A_{41} = A_{43} = A_{45} = A_{47} = 1$$

2. Count the nonzeros: nine diagonal entries and $4 \cdot 3 \cdot 2 = 24$ neighbour entries.

$$5m^2 - 4m = 45 - 12 = 33$$

3. Ap is the first column of A: -4 in cell 0 and $+1$ in its neighbours 1 and 3.

$$Ap = (-4, 1, 0, 1, 0, 0, 0, 0, 0)^T$$

Answer. 33 nonzeros; $A p = (-4, 1, 0, 1, 0, 0, 0, 0, 0)$. The Matrix tab at $m = 3$ shows the same 9 by 9 pattern, and `tests/js/flow_operator_test.mjs` checks the solver's operator against a dense matrix written from the rule for seven grids to $1e-12$.

Analyze

1. Explain why a box with walls on all four sides gives a singular matrix, and why the solution is then only defined up to a constant. *Hint: Add up the entries of a row at an interior cell, an edge cell and a corner cell.*
2. The Matrix tab shows heavy lines every m rows and columns. What do they separate, and why are the entries just outside the central blocks the vertical neighbours? *Hint: With line-by-line numbering, the cell above is m positions later.*
3. Why can a checkerboard pressure (every other cell $+1$ and -1) be present in the solution of $Ap = b$ and yet have no effect on the velocity update? *Hint: Take the central difference of $(-1)^{i+j}$ over two cells.*

Evaluate

1. Would you ever form the 4096 by 4096 matrix of the 64 by 64 grid explicitly? Compare the memory of a dense matrix with a sparse list. *Hint: A dense float64 matrix has 4096 squared entries, 134 MB; the 20,224 nonzeros take about 243 kB with an index each.*
2. A collocated grid is simple but leaves a floor of divergence. A staggered grid removes it. What would you give up in the hero by switching? *Hint: Think of the shader code: how many textures, how many neighbour reads, how boundary conditions at an obstacle would have to be written.*

Create

1. Add an obstacle (a solid block) to the matrix view and predict which rows change and which entries disappear before drawing it.
2. Design a check that distinguishes an operator with the wrong boundary rule from one with the wrong stencil, using the property $A\mathbf{1} = 0$.

Check yourself

1. understand: What does one row of the pressure matrix say, in words?
Hint: Sum over neighbours minus four times the cell.
Answer: That the pressure of a cell, compared with the average of its four neighbours, is set by the divergence of the predicted velocity in that cell: sum of the neighbours minus four times the cell equals the divergence. Pressure is smooth except where fluid is being created or destroyed.
2. apply: How many nonzeros does the matrix of the 64 by 64 open grid have, and what fraction of the 4096 by 4096 entries is that?
Hint: $5m^2 - 4m$.
Answer: 20,224 nonzeros, 0.121 % of the entries (for $m = 6$: 156 of 1296, 12.04 %). A dense copy would need 134 MB.
3. analyze: On a 6 by 6 grid with walls on every side, what is the trace of A , and why is A singular?
Hint: Count interior, edge and corner cells.
Answer: 16 interior cells with diagonal -4 , 16 edge cells with -3 and 4 corners with -2 : trace = $-64 - 48 - 8 = -120$ (the open grid has -144). Each row sums to zero, so the all-ones vector satisfies $Ax = 0$ and A has no inverse.
4. evaluate: For the plume field the divergence after the projection is 0.51 of the divergence before for 80 Jacobi sweeps and 0.52 for an exact CG solve. How can the less converged solve be better?

Hint: What does the solve minimise, and what is the divergence measured with?

Answer: The solve drives the residual of the compact five-point system to zero. The divergence after the projection is measured with the wide central-difference stencil. For a rough right-hand side the two are different quantities, so minimising one does not minimise the other, and an unconverged pressure can happen to leave slightly less wide divergence. The difference (0.507 against 0.519) is small compared with the floor of about 0.5 that neither can remove.

5. create: Sketch a modification of the projection that has no floor on a collocated grid.

Hint: Make the operator in the solve equal the divergence of the gradient that is used for the update.

Answer: Solve with the wide operator itself: build A as D G, where G is the central-difference gradient and D the central-difference divergence, so that the update removes exactly the divergence the solve targets. The price is that the matrix couples cells two apart, splits into four independent checkerboard-like subgrids and is singular on the checkerboard modes. Rhie-Chow interpolation and staggered grids are the usual remedies.

Numbers on this page and where they come from

Quantity	Value	Source
Nonzeros of the open $m \times m$ matrix, $m = 6$ ($n = 36$) and $m = 64$ ($n = 4096$)	156 and 20,224	tools/flow_numbers.py (numpy / sympy) ($5m^2 - 4m$; the dense matrix agrees: 156)
Fraction of nonzero entries, $m = 6$ and $m = 64$	12.04 % and 0.121 %	tools/flow_numbers.py (numpy / sympy)
Dense and sparse storage for $m = 64$	134.2 MB and 243 kB	tools/flow_numbers.py (numpy / sympy)
Condition number, open grid, $m = 6$	19.196	tools/flow_numbers.py (numpy / sympy) (numpy cond: 19.196)
Condition number, open grid, $m = 64$	1711.7	tools/flow_numbers.py (numpy / sympy) ($\cot^2(\pi/130)$)
Trace of A on 6 by 6: walls on all sides, open sides	-120 and -144	tools/flow_numbers.py (numpy / sympy)
Wide divergence of the gradient of a checkerboard, and $ A $ of it	0 and 8	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js)
Plume field, divergence after / before: Jacobi 6, 24, 80 sweeps	0.546, 0.508, 0.507	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js), $N = 64$, step 150, warm start 0.8
Plume field, divergence after / before: exact CG (600 iterations)	0.519	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js)
Smooth field ($N = 32$), exact solve: divergence after / before	0.00241	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js)
Solver operator against a dense reference matrix, relative error	0 to $1e-12$ (seven grids)	tests/js/flow_operator_test.mjs

Sources

- Trefethen, L. N.; Bau, D. (1997). Numerical Linear Algebra. SIAM.

5 Vorticity confinement and buoyancy are forces added by hand: the first puts back the curls the grid smears out, the second makes hot gas rise

Numerical diffusion damps small vortices, so a simulated plume looks smoother than a real one. Fedkiw's vorticity confinement computes where the vorticity is concentrated and adds a force that pushes the flow around those places, with a strength set by one parameter. Buoyancy is a body force proportional to the temperature (and minus the weight of the smoke), the Boussinesq model. Both are cheap and visually effective, and both are modelling choices, not physics.

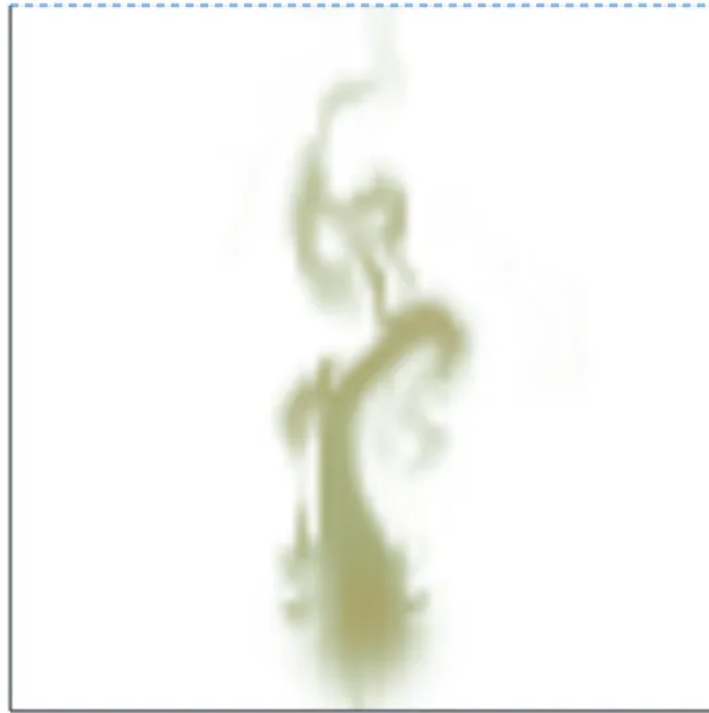


Figure 5: Screenshot of the Flow workspace (scene 0): a hot plume driven by buoyancy, with vorticity confinement keeping its curls.

Read the figure

Question. Do the curls of the plume come from the physics or from the added forces, and what happens if you switch the forces off?

Axes. The picture is the simulation box, x to the right and y up. Brightness of teal is the dye, amber is the temperature (hot gas). The bottom-centre bright spot is the burner that injects hot dye and an upward jet.

Look at.

- The mushroom heads at the top of the plume: that is a pair of counter-rotating vortices lifted by the buoyancy.
- The width of the stem: set by the burner size and by how much the grid diffuses it.
- What happens to the small curls when Vorticity confinement is set to 0 in the panel.

Must check.

- The confinement strength and the grid (the force scales with the cell size).

- The buoyancy and weight coefficients and the temperature scale.
- Whether any dissipation of the temperature and dye is on.

If it is wrong.

- Reading the crisp curls as a resolved turbulence result when they are the result of a tuned force.
- Raising the confinement until the picture looks right and then trusting the energy or the vorticity numbers.

What it proves. That the flow keeps its small-scale rotation at the chosen strength. It does not prove that the plume has the physical curl size: confinement adds vorticity beyond what the inviscid flow would keep.

Terms

Vorticity $\omega = \partial_x v - \partial_y u$ The local spin of the flow: twice the angular velocity of a fluid parcel. In two dimensions it is a single number per point.

Circulation $\Gamma = \oint u \cdot dl = \iint \omega dA$ The line integral of the velocity around a closed loop, equal to the vorticity integrated over the area enclosed.

Vorticity confinement $f_c = \varepsilon h (N \times \omega \hat{z})$ A force that points around the local maxima of the vorticity magnitude, strengthening the spin that numerical diffusion removes.

Gradient of the vorticity magnitude $N = \frac{\nabla|\omega|}{|\nabla|\omega||}$ The direction in which the vorticity magnitude grows fastest. Normalised, it points toward the vortex centre.

Boussinesq approximation $f = -\frac{\rho - \rho_0}{\rho_0} g \hat{y} \approx \beta g \Delta T \hat{y}$ Treat the density as constant except in the gravity term, where its small variation drives the flow.

Thermal expansion coefficient $\beta = \frac{1}{T}$ The fractional change of density per kelvin. For an ideal gas at temperature T it is 1/T.

Kelvin-Helmholtz instability Two layers sliding past each other roll up into vortices. The interface between them is a vortex sheet.

Passive dye $\partial_t c + u \cdot \nabla c = 0$ A tracer that is carried by the flow and does not change it. In the plume the dye is also given a weight, which makes it slightly non-passive.

Remember

- Vorticity $\omega = \partial_x v - \partial_y u$; in the workspace it is stored as the central difference $\frac{1}{2}(v_R - v_L - u_T + u_B)$, which is $h \omega$.
- Confinement: $f_c = \varepsilon h (N_y, -N_x) \omega$ with $N = \nabla|\omega|/|\nabla|\omega||$. It adds energy; it does not conserve circulation. Here ε is the hero's value at 128 cells, rescaled by $N/128$ on this grid.
- Buoyancy: $\Delta v_y = \Delta t (b T - w c)$ with temperature T and dye density c ; Boussinesq for air: acceleration = $g \Delta T / T_0$.
- Air near 300 K: 0.0327 m/s² per kelvin, so a gas 10 K warmer accelerates at 0.327 m/s² and one 100 K warmer at 3.27 m/s².
- An ideal vortex sheet of velocity jump ΔU grows with rate $\sigma = k \Delta U / 2$ at wavenumber k ; a smooth shear layer of thickness δ grows more slowly.
- Numerical diffusion damps vorticity; confinement can put back more than it took (see the numbers below), so ε has to be tuned, never trusted.

Understand

Easy: put the spin back Imagine stirring coffee: the swirls fade because the liquid is a little sticky. A simulation on a grid makes the liquid artificially sticky, so small swirls fade far too fast and the smoke looks like soft cloth. Vorticity confinement finds where the spinning is strongest and gives it a little push in the direction it is already spinning, a kind of anti-stickiness. Buoyancy is simpler: hot gas is lighter, so we push it up in proportion to its temperature.

Medium: the force in three steps Compute the vorticity ω . Find its magnitude $|\omega|$ and the direction N in which that magnitude grows: it points to the centre of the nearest vortex. The force is perpendicular to N and proportional to ω , so it circles the centre in the sense of the spin: $f = \varepsilon h (N_y, -N_x) \omega$. The strength ε is a free parameter; the cell size h in the formula makes the force go to zero as the grid is refined, as it should.

Hard: it fights diffusion, but it is not a conservation law Run the Shear layer scene for 100 steps with confinement 0, 3 and 6. The initial peak vorticity is 12.55 per second. After 100 steps (1.67 s) it is 10.34 without confinement (diffused), 29.0 with the slider at 3 and 49.2 at 6. An inviscid flow cannot raise the peak above its initial value in two dimensions, so the confined runs contain more vorticity than physics allows: confinement is an artistic force, which is the price for sharpness.

Hardest: what the numbers say about the instability and the plume An infinitely thin vortex sheet of velocity jump 1 and wavelength 0.5 (the scene's two waves) would grow at $\sigma = k\Delta U/2 = 6.28$ per second (e -folding time 0.159 s). The scene's shear layer has thickness 0.04 ($k\delta = 0.50$) and the standard result for a tanh layer gives a maximum growth rate of about $0.19 U/\delta = 2.37$ per second ($U = 0.5$, $\delta = 0.04$); the solver's growth of the largest cross-flow speed between 0.5 s and 1 s was 1.52, 2.32 and 1.54 per second for confinement 0, 3 and 6: the right order of magnitude, noisy because it is a two-point estimate of a growing mode that is still emerging. In the plume, the burner's temperature 0.82 with buoyancy 4.2 and dye weight 0.25 on dye 0.55 gives a net upward acceleration of 3.31 domain lengths per second squared; from rest the plume head would rise half a box in about 0.55 s if there were no drag, which is the right scale for what the scene shows.

Derivation

Step 1. Vorticity from the velocity

$$\omega = \partial_x v - \partial_y u \approx \frac{(v_R - v_L) - (u_T - u_B)}{2h}$$

The workspace stores the numerator over two (central differences), which is $h \omega$. It is the same pass as the hero's curl shader.

Step 2. Where is the vortex centre?

$$N = \frac{\nabla|\omega|}{|\nabla|\omega|| + 10^{-4}}, \quad \nabla|\omega| \approx \frac{1}{2}(|\omega_R| - |\omega_L|, |\omega_T| - |\omega_B|)$$

The gradient of the vorticity magnitude points up the hill toward the strongest spin. The small constant keeps the division finite.

Step 3. The confinement force

$$f_c = \varepsilon h (N_y, -N_x) \omega, \quad u \leftarrow u + \Delta t f_c$$

$N \times \omega \hat{z}$ in two dimensions. It is tangent to the circles around the centre and its sign follows the spin, so it speeds the swirl up. Vorticity of either sign is treated alike.

Step 4. Buoyancy from the Boussinesq model

$$\rho = \rho_0(1 - \beta \Delta T) \Rightarrow f = \beta g \Delta T \hat{y}, \quad \beta = \frac{1}{T_0}$$

For air at 300 K: 0.327 m/s² for a gas 10 K warmer. The workspace uses a dimensionless temperature and a free coefficient b .

Step 5. The step in the workspace

$$v_y \leftarrow v_y + \Delta t (bT - wc), \quad b = 4.2, w = 0.25$$

Temperature T pulls the gas up, the dye density c weighs it down, as in the hero's force shader.

Step 6. Linear stability of a vortex sheet, for scale

$$\sigma = \frac{k \Delta U}{2} \text{ (thin sheet)}, \quad \sigma_{\max} \approx 0.19 \frac{U}{\delta} \text{ (tanh layer)}$$

The thin-sheet value is an upper bound for any thickness; a finite layer is slower and has a cut-off wavelength.

Worked example: How fast does a hot parcel rise, and how long for the plume head?

Air near 300 K carries a parcel 10 K warmer than its surroundings. Find its initial upward acceleration with the Boussinesq model. Then, for the Smoke plume scene, find the net acceleration of the burner gas (temperature 0.82, dye 0.55, buoyancy 4.2, weight 0.25) and the time to rise half a box from rest without drag.

1. Thermal expansion coefficient of an ideal gas, then the acceleration.

$$\beta = \frac{1}{300} \text{ K}^{-1}, \quad a = g\beta\Delta T = 9.81 \times \frac{10}{300} = 0.327 \text{ m/s}^2$$

2. In the scene, add the buoyant and the weight terms.

$$a = bT - wc = 4.2 \times 0.82 - 0.25 \times 0.55 = 3.3065$$

3. Constant acceleration from rest covers a distance $\frac{1}{2}at^2$; set it to 0.5.

$$t = \sqrt{\frac{2 \times 0.5}{3.3065}} = 0.550 \text{ s}$$

Answer. $a = 0.327 \text{ m/s}^2$ for the 10 K parcel; in the scene $a = 3.306$ domain lengths per second squared and the rise time to half a box is 0.55 s, before drag and mixing slow it down.

Analyze

1. Show that the confinement force is perpendicular to N , and explain why that makes it circle the vortex centre instead of pushing toward or away from it. *Hint: $f \propto (N_y, -N_x)$; take the dot product with $N = (N_x, N_y)$.*
2. The force scales with the cell size h . What would happen on a very fine grid if it did not, and why does that make the confinement a consistent modification? *Hint: As $h \rightarrow 0$ the numerical diffusion also goes to zero, so the correction should vanish too.*
3. Why is the buoyancy of a dense, dye-heavy parcel handled by subtracting a weight term instead of letting the temperature alone do the job? *Hint: Smoke is denser than clean air at the same temperature; the dye term stands in for the extra mass.*

Evaluate

1. Confinement 6 raised the peak vorticity of the shear layer to 49 per second from an initial 12.6. A student argues that the simulation is more accurate because the vortices are sharper. Evaluate the argument. *Hint: In 2D inviscid flow the maximum of the vorticity cannot grow.*
2. For which applications is confinement acceptable and for which not? Give one of each. *Hint: A film effect needs the right look; a measurement of the kinetic energy decay of turbulence needs the right physics.*

Create

1. Design a test that separates the numerical vorticity loss of the scheme from the vorticity added by confinement, for example with a single vortex in a periodic box.
2. Propose a buoyancy model in which the temperature also diffuses and cools with height, and say which parameters you would add to the panel.

Check yourself

1. understand: What does vorticity confinement do, and why is it needed?
Hint: Numerical diffusion.
Answer: It finds the local maxima of the vorticity magnitude and adds a force that circles around each one, strengthening the spin. It is needed because advection on a grid acts like an artificial viscosity that damps small vortices much too fast, so the simulated smoke looks smoother than real smoke.
2. apply: What is the Boussinesq acceleration of a parcel of air 10 K warmer than its surroundings at 300 K?
Hint: $g \beta dT$ with $\beta = 1/T$.
Answer: $a = 9.81 \times 10 / 300 = 0.327 \text{ m/s}^2$.
3. analyze: After 1.67 s the shear layer's peak vorticity was 10.3, 29.0 and 49.2 per second for confinement 0, 3 and 6, starting from 12.6. Interpret the three numbers.
Hint: Compare each with the starting value.
Answer: Without confinement the grid has diffused the peak to 82 % of its initial value. With confinement the peak is 2.3 and 3.9 times the initial: more than the inviscid flow can keep, so the force is adding vorticity (partly by concentrating it as the layer rolls up, partly artificially).

4. evaluate: A plume simulation looks right with confinement 3 on a 64 by 64 grid and too noisy on a 96 by 96 grid at the same slider value. Is something wrong with the code?

Hint: Compare the peak vorticity of the shear scene at the two resolutions with the same slider.

Answer: Not necessarily. The workspace rescales the hero's value by $N/128$, yet after 1.67 s the shear layer's peak vorticity at slider 3 was 29.0 per second at 64 cells and 40.9 at 96 cells: the response of the flow depends on the resolution. The honest procedure is to retune epsilon per resolution and to treat it as a visual parameter, never as a physical constant.

5. create: Design an experiment in the Flow workspace that shows confinement adds energy, not just sharpness.

Hint: Watch the kinetic energy in the Dashboard.

Answer: Use the Shear layer scene with viscosity and dissipation at 0 and compare the kinetic energy readout after the same time for confinement 0, 3 and 6. The solver gave 0.1152 to 0.1129 (a loss of 2.0 %) without confinement, 0.1153 to 0.1159 at 3 and 0.1154 to 0.1199 at 6 (a gain of 3.9 %). An inviscid periodic flow cannot gain energy, so the gain is what the force added.

Numbers on this page and where they come from

Quantity	Value	Source
Ideal-gas thermal expansion coefficient at 300 K	0.00333 per K	1/T
Boussinesq acceleration for 1 K, 10 K and 100 K warmer ($g = 9.81$)	0.0327, 0.327, 3.27 m/s ²	tools/flow_numbers.py (numpy / sympy)
Plume scene: net acceleration b T - w c (4.2 x 0.82 - 0.25 x 0.55)	3.3065	tools/flow_numbers.py (numpy / sympy)
Plume scene: time to rise 0.5 from rest without drag	0.550 s	tools/flow_numbers.py (numpy / sympy)
Shear layer: initial peak vorticity	12.55 per s	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js)
Peak vorticity after 100 steps: confinement 0, 3, 6	10.34, 29.02, 49.20 per s	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js), $N = 64$, $dt = 1/60$ s, 40 pressure iterations
Kinetic energy, first step to step 100: confinement 0, 3, 6	0.1152 to 0.1129; 0.1153 to 0.1159; 0.1154 to 0.1199	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js), $N = 64$
Peak vorticity after 100 steps, confinement 3, at $N = 96$	40.9 per s	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js), $N = 96$
Enstrophy per cell after 100 steps: confinement 0, 3, 6	0.00163, 0.00302, 0.00654	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js) (vorticity in per-cell units)
Thin vortex-sheet growth rate $k dU / 2$ ($dU = 1$, wavelength 0.5)	6.283 per s	tools/flow_numbers.py (numpy / sympy)

Quantity	Value	Source
tanh-layer maximum growth rate 0.1897 U / delta (U = 0.5, delta = 0.04)	2.371 per s	tools/flow_numbers.py (numpy / sympy) (standard linear-stability coefficient)
Measured growth of max v , 0.5 s to 1 s, confinement 0, 3, 6	1.52, 2.32, 1.54 per s	tools/flow_numbers.mjs: the Flow workspace's own solver (flow_core.js) (a two-point estimate, noisy)
Effective confinement at N = 64 for the slider value 3	1.5	3 x N / 128

Sources

- White, F. M. (2011). Fluid Mechanics. McGraw-Hill.