Upwind is used because the equation contains *advection* in x and z, and an explicit scheme must control numerical instability from drift. Leapfrog is a *centered, second-order, nondissipative* method. For advection it is *unconditionally unstable unless paired with special staggering* (Yee grid / Lax–Wendroff stabilization). A 3D FP code with general drifts cannot use leapfrog safely. Upwind guarantees monotonicity and positivity of the advected density; leapfrog does not.

Implicitness enters only where the stability restriction is severe, i.e. diffusion in ϕ. The advection CFL is typically mild; diffusion CFL is the tight constraint. This is why Chang–Cooper is often embedded in a *semi-implicit* update:

- Explicit advection (cheap, first-order).
- Implicit or semi-implicit diffusion (removes the $\Delta t \leq \Delta\phi^2/(2D_0)$ limit).

Full implicitness requires solving a *3D linear system* of size $N_x N_z N_\phi$; this is computationally inefficient unless the grid is extremely small. The operator is not separable when drifts depend on the state, and one loses the simplicity of the flux form.

The only part where implicitness is rational:

- diffusion in ϕ, because it is one-dimensional along axis 2 and leads to *tridiagonal systems for each* (x, z), solvable by the Thomas algorithm at negligible cost.

Applying the same logic to advection is not useful: implicit upwind offers no meaningful advantage, and implicit centered schemes require limiters or TVD corrections, destroying the benefit of the flux form.

Minimal viable improvement:

- Keep x, z advection *explicit upwind*.
- Keep Chang–Cooper for drift in ϕ.
- Replace the explicit diffusive term

$$-D\,\frac{f_{k+1} - 2f_k + f_{k-1}}{\Delta\phi^2}$$

  with an *implicit* (or Crank–Nicolson) tridiagonal solve along ϕ.

This relaxes the diffusion CFL while preserving positivity and the stationary distribution. Full implicitness is computationally wasteful; semi-implicit treatment in ϕ is optimal.