Implementing Level Sets

$\boxed{L43}$

Goal: To find out how to numerically solve the level-set evolution equation.

Recall the PDE that governs the evolution of the level-set embedding function,

$$\frac{\partial \phi}{\partial t} = -V_N \|\nabla \phi\|$$

Time Derivative

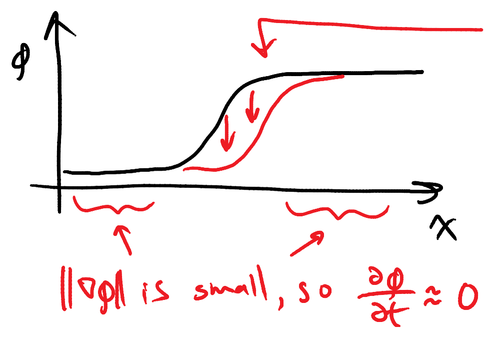
We will use **forward** differencing to give us an explicit time-stepping method.

$$\frac{\phi^{t+1} - \phi^t}{\Delta t} \cong \frac{\partial \phi}{\partial t} \Rightarrow \phi^{t+1} = \phi^t - \Delta t \, V_N^t \|\nabla \phi^t\|$$

Spatial Derivative

Estimating the gradient of ϕ turns out to be more problematic. There are important issues to keep in mind in order for the numerics to give good results.

Consider a wave going from left to right

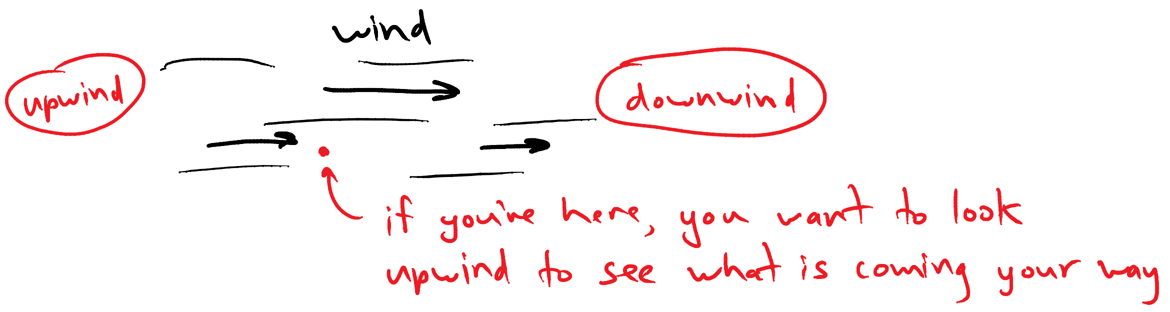


ϕ
x
$\|\nabla\phi\|$ is small, so $\frac{\partial \phi}{\partial t} \approx 0$

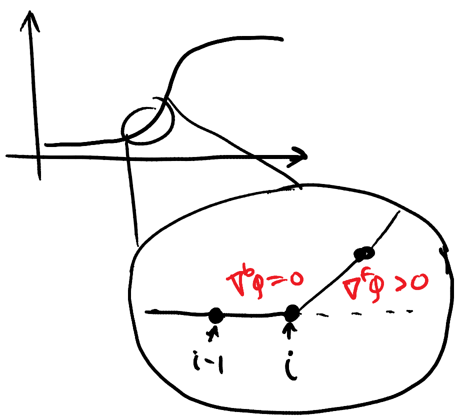
We need $\frac{\partial \phi}{\partial t} < 0$ here.

$\therefore$ To move to the right, $V_N > 0$.

If our wave is travelling to the right, then we should look **"upwind" (left)** when computing our derivatives.


wind
upwind
downwind
if you're here, you want to look upwind to see what is coming your way

Let $\nabla^b \phi$ be the backward diff. approx. of $\frac{\partial \phi}{\partial x}$ (or $\nabla \phi$), and $\nabla^f \phi$ be the forward diff. approx.

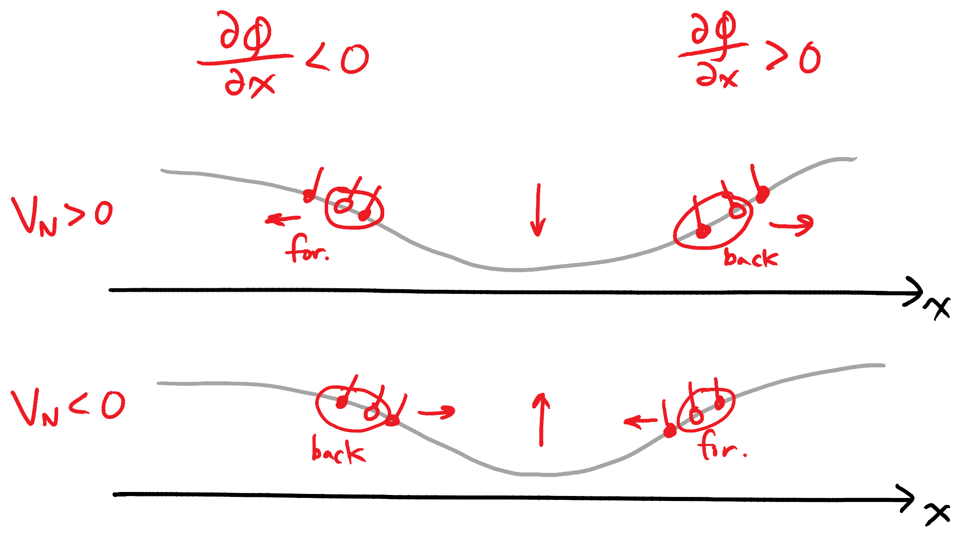


Using back. diff. $\nabla^b \phi = 0$

$\Rightarrow$ no change in ϕ_i (smooth landing)

Using forward diff. $\nabla^f \phi > 0$

$\Rightarrow$ ϕ_i drops down (overshoot)


$\frac{\partial \phi}{\partial x} < 0$
$\frac{\partial \phi}{\partial x} > 0$
$V_N > 0$
for.
back
x
$V_N < 0$
back
for.
x

$$\nabla^+ = \min(\nabla^f \phi, 0) + \max(\nabla^b \phi, 0)$$

$$\nabla^- = \min(\nabla^b \phi, 0) + \max(\nabla^f \phi, 0)$$

Implementation

- Start with an initial embedding function ϕ

- Compute speed function from image

$$\text{grad\_f} = \text{gradient}(f, \text{'central'})$$
$$V = \frac{1}{1 + \|\text{grad\_f}\|^2}$$

- Compute partial derivatives of ϕ using both forward and backward differencing

$$\frac{\partial \phi}{\partial x} \begin{cases} \nabla_x^f = \text{MyDerivative}(\phi, x, \text{'forward'}) \\ \nabla_x^b = \text{MyDerivative}(\phi, x, \text{'backward'}) \end{cases}$$
$$\frac{\partial \phi}{\partial y} \begin{cases} \nabla_y^f = \ldots \\ \nabla_y^b = \end{cases}$$

- Combine like partial derivatives, choosing forward or backward as appropriate

$$\nabla_x^+ = \min(\nabla_x^f, 0) + \max(\nabla_x^b, 0)$$
$$\nabla_x^- = \max(\nabla_x^f, 0) + \min(\nabla_x^b, 0)$$

Likewise for ∇_y^+ and ∇_y^-.

- Choose upwind derivative, according to speed function

$$\nabla\phi(x_y) = \begin{cases} (\nabla_x^-, \nabla_y^-) & \text{if } V(x_y) \le 0 \\ (\nabla_x^+, \nabla_y^+) & \text{if } V(x_y) > 0 \end{cases}$$

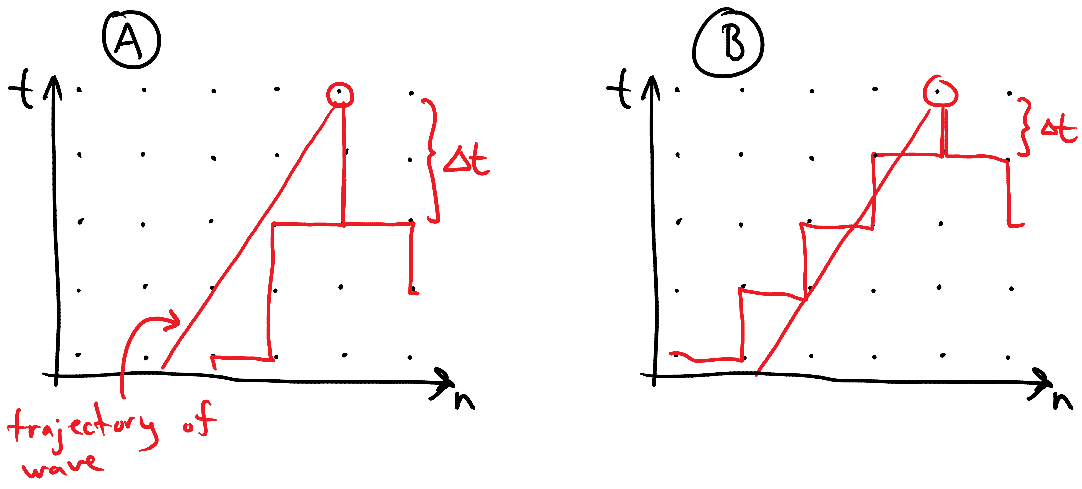
- Take time step

$$\phi^{t+1}(x,y) = \phi^t(x,y) - \Delta t\, V(x,y)\, \nabla\phi^t(x,y)$$

CFL Condition (Courant-Friedrichs-Lewy Condition)

We need to make sure that our time steps are not too big. As the "wind blows", we must make sure that the information keeps pace with our computations.

eg. Computing ϕ_n^{t+1} involves ϕ_n^t, ϕ_{n-1}^t and ϕ_{n+1}^t



As you go back in time, **A** shows that the basin of influence **does not include** the wave, so will be unstable. In **B**, the basin of influence **includes the wave,** so will be more dependable.

In general, you must choose a Δt small enough so that the **fastest-moving** things do not move more than **one spatial sample per step.**

$$V_{max} \frac{\Delta t}{\Delta x} \leq 1$$ for explicit time stepping.

— END