Consider an advection-diffusion equation with BC:

\[ u_t + a u_x = d u_{xx}, \quad t \geq 0 \]
\[ 0 < x < L = 1 \]

\[ \begin{cases} u(0, t) = 1 \\ u(1, t) = 0 \quad \text{Dirichlet} \\ u_x(1, t) = 0 \quad \text{Neumann} \end{cases} \]

For Dirichlet, the steady-state solution \( u_t = 0 \) \( \Rightarrow \) \( a u_x = d u_{xx} \)

\[ u(x, t) = \left[ e^{a/d} - e \right] / \left[ e^{a/d} - 1 \right] \]
Boundary layer of width $\sim \frac{d}{a}$

The solution forgets the boundary condition after a distance $\Delta x \sim \frac{d}{a}$ away from the boundary. Of

$$Pe = \left( \frac{\Delta x}{L} \right)^{-1} = \frac{aL}{d} \gg 1$$

This layer will be very thin! (Compare to airplane wing)
If we used Neumann BC, however, we would get

\[ u(x, t) = 1 \]

with no boundary layer.

This is what we often do in CFD if it is not important to resolve the boundary layer. We make it disappear by using a fake boundary condition (extrapolation from interior), as we will see next.

For now, assume smooth solution!
Consider first advection-dominated flow:
\[ u_t + u_x = 0, \quad x \in (0,1) \]
\[ u(0, t) = g_0(t) \]
no BC at outflow boundary

Every CFD code has them!

Assume we use centered difference in the interior

"ghost" cell or "virtual" cell
\[
\frac{w'}{j} = \frac{1}{2h} (w_{j-1} - w_{j+1})
\]

At the left boundary,
\[
w_0(t) = \varphi_0(t)
\]
follows from BC.

At the right boundary, naive application of the same stencil leads to
\[
w'_m = \frac{1}{2h} (w_{m-1} - w_{m+1})
\]
but \(w_{m+1}\) is not a real variable.

Since there is no BC at right, the best we can do is extrapolate from the interior.
\[ w_{m+1} = w_m - \text{constant extrapolation} \]

which leads to \[ w'_m = \frac{1}{2h}(w_{m-1} - w_m) \]

or linear extrapolation \[ w_{m+1} = 2w_m - w_{m-1} \] \[ \Rightarrow \]

\[ w'_m = \frac{1}{2h} \left[ w_{m-1} - 2w_m + w_{m+1} \right] \]

\[ w'_m = \frac{1}{h} (w_{m-1} - w_m) \]

which means we are using the upwind scheme at the right boundary.

This makes sense!
So let us choose

\[ W_{m}^{'} = \frac{1}{h} (W_{m-1} - W_{m}) \]

which we can either implement in the code as a one-sided special stencil at the boundary, or, we can keep the same stencil but use a ghost cell

\[ W_{m+1} = 2W_{m} - W_{m-1} \] (better?)

The problem we have now is that we switched from a second order to first order spatial local truncation error near the boundary.
We can prove that the scheme is still second-order accurate!

Denote the local spatial truncation error with $\delta_h$ as before, and consider

$$w' = A w + g(t)$$

and assume that

$$\delta_h = A \xi(t) + \eta(t)$$

Then, the global error

$$e' = A e + \delta_h = A (e + \xi) + \eta$$

Take norms after variation of constants exactly as before, to get
\[ E = E + \xi \]

\[ E' = A E + 2 \xi + \eta (t) \]

\[ \| E(t) \| \leq \| \xi \| + Ke^{wt} \| E(0) + \xi(0) \| + \frac{K}{w} (e^{wt} - 1) \max_{0 \leq s \leq t} \| E'(s) + \eta(s) \| \]

This means that

**Theorem:** If \[ \| \xi \|, \| \xi' \|, \| \eta \| \leq Ch^r \]

and \[ \| E(0) \| \leq C_0 h^r \], then

\[ \| E(t) \| \leq C h^r \]

\[ \leq \text{global order } = r \]

\[ \leq \text{local order } = r \]
The difference with before is that this requires a bound on \( \| \mathbf{g}(t) \| \) and \( \| \mathbf{f}'(t) \| \) and \( \| \mathbf{N}(t) \| \) \textit{NOT} on \( \| \mathbf{6h}(t) \| \). Recall

\[
\mathbf{6h} = A \mathbf{g} + \mathbf{N} \Rightarrow \mathbf{g} = A^{-1} \mathbf{6h}
\]

Note that \( \mathbf{6h} \) is the solution of the \underline{steady-state} problem with forcing

\[
\mathbf{g} = \mathbf{N} - \mathbf{6h}
\]

So the local error acts as a source term that then propagates through the \underline{global solution}, spatially.
Let's now go back to our example:

\[ A = \frac{1}{2h} \begin{bmatrix} 0 & -1 \\ -1 & 0 \\ -1 & -1 \\ 1 & 0 & -1 \\ 2 & -2 \end{bmatrix} \]

\[ \begin{aligned} \delta h, m = & \ U_t(x_m, t) - \frac{1}{h} \left[ U(x_{m-1}, t) - U(x_m, t) \right] \\ = & \ - \frac{1}{2} h U_{xx}(x_m, t) + O(h^2) \end{aligned} \]

Assume smoothness \( \| U_{xx} \| \leq c_1 \)

\[ \Rightarrow \delta h, m \leq c_1 h \]

\[ \delta h, j < m \leq c_2 h^2 \]
So we have

\[ \delta_h = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} c_1 h + O(h^2) \]

and we need to solve

\[ \delta_h = A \xi + \eta \Rightarrow \]

\[ \frac{1}{2h} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \xi = \begin{bmatrix} 0 \\ 1 \end{bmatrix} c_1 h \]

This system we can solve by hand

\[ \xi = \begin{bmatrix} \ldots, \xi_m, 0, \xi_m, 0, \xi_m \end{bmatrix} \]
\[ S_m = C_1 h^2 \]

So indeed \( \| S \| = O(h^2) \), \( \| \eta \| = O(h^2) \) and the theorem tells us the spatial semi-discretization is second-order accurate!

This is a very common occurrence, and we often do low-order discretizations near boundaries, even tenth-order (!?) (inconsistent) stencils next to boundaries. The hard part is stability!
There is no simple general way to analyze stability, i.e., to show that
\[ \| e_{t+h} \| \leq K e^{\omega t} \]
uniformly in \( h \).

The wisdom is that for schemes with some (artificial or real) dissipation,
\[ \text{stability is ensured by damping of any instabilities near the boundaries.} \]

Often we rely on numerical experiments, since instabilities are typically obvious.
For finite-volume schemes often boundaries overlap with the faces of the grid, i.e., half-integer points in 1D.

\[ X_j = (j - \frac{1}{2})h, \quad h = 1/m \]

\[ u_t = u_{xx} \quad \text{with BC} \begin{cases} u(0,t) = f_0(t) \\ u(1,t) = g_1(t) \end{cases} \]

We need ghost values \( u_0 \) and \( u_{m+1} \).
Left:

\[ \frac{W_0 + W_1}{2} = \delta_0 \leq \text{linear ext/interpolation of BC} \]

\[ \Rightarrow \begin{cases} W_0 = 2\delta_0 - W_1 \end{cases} \]

Right: What Neumann BC really means is no-flux through \( x = 1 \):

\[ F(x = 1) = \frac{W_{m+1} - W_m}{\Delta x} = 0 \Rightarrow \]

\[ W_{m+1} = W_m \]

(constant extrapolation)
New consider advection - diffusion

with finite volume

\[ u_t + a u_x = d u_{xx} \]

Dirichlet BC gives flux directly for advection, \( f = a n \)

Neumann BC gives flux directly for diffusion, \( f_{\text{diff}} = -d u_x \)

So for those fluxes we do not need any interpolation/extrapolation or ghost cells!
If we want our method to work for \( d \rightarrow 0 \) or advection-dominated flows, we must treat advection separately and realize that there is no BC at outflow for advection, only for diffusion.

So treat diffusion as already explained in lecture notes.

For advection, we need to consider second or third-order schemes separately.
\begin{align*}
\{ \quad & u(0,t) = \tilde{f}(t) \quad \text{Dirichlet on left} \\
& u_x(1, L_x, t) = \tilde{g}(t) \quad \text{for diffusion but does not matter for advection!} \\
\text{At the right, we have outflow boundary} \\
\text{and at the left,} \\
\text{inflow boundary} \\
\frac{\partial u}{\partial t} = f(0) = a \tilde{f}(t) \\
\text{(flux at } x = 0) \\
\tilde{f}_{1/2} = \tilde{f}(h) \\
\tilde{f}_{3/2} \approx \tilde{f}(x = h) = ?
\end{align*}
An obvious choice is

1. **Centered - advection:**

\[
\frac{f}{3^{1/2}} = -a \frac{w_1 + w_2}{2} \quad (\text{no ghost cell})
\]

This gives (this is crucial to check)

\[
\frac{dw_1}{dt} = -\frac{a}{h} \left( w_1 + w_2 - w_{1/2} \right)
\]

where

\[
w_{1/2} = U(x=0) = \hat{f}(t)
\]

A Taylor series expansion shows that this is first-order accurate at the boundary, which is on since we get +1 order from 1/2 in stencil.

**Note:** One can pretend here this is finite-difference...
At the outflow boundary, we want a one-sided stencil (no BC) so it is most natural to use first-order upwinding at outflow.

\[ \frac{dW_n}{dt} = -\frac{\alpha}{h} (W_n - W_{n-1}) \]

Simple algebra shows that this is the same as using a ghost cell with linear extrapolation.

(Must check this!)

\[ W_{n+1} = 2W_n - W_{n-1} \]