EPFL

LFMI

The finite volume method

Numerical Flow Simulation

Edouard Boujo

Fall 2022

École
polytechnique
fédérale
de Lausanne

Governing equations: incompressible flows

- Unknowns $\mathbf{u} = (u, v, w),\ p,\ T$ functions of $\mathbf{x},\ t$

- Conservation of mass (continuity equation)

$$\nabla \cdot \mathbf{u} = 0$$

- Conservation of momentum

$$\frac{\partial(\rho\mathbf{u})}{\partial t} + \nabla \cdot (\rho\mathbf{u}\mathbf{u}) = -\nabla p + \nabla \cdot (\mu\nabla\mathbf{u}) + \rho\mathbf{f}$$

μ dynamic viscosity [kg/(m.s)]

$\mathbf{f}$ body force [m/s^2]

- Conservation of energy

$$\frac{\partial(\rho T)}{\partial t} + \nabla \cdot (\rho T\mathbf{u}) = \nabla \cdot (\Gamma\nabla T) + \frac{S}{c_p}$$

$\Gamma = k/c_p$

k thermal conductivity [W/(m.K)]

c_p specific heat capacity [J/(kg.K)]

S source [W/m^3]

Numerical Flow Simulation

Governing equations: incompressible flows

- Unknowns $\mathbf{u} = (u, v, w),\ p,\ T$ functions of $\mathbf{x},\ t$

- Conservation of mass (continuity equation)

$$div(\mathbf{u}) = 0$$

- Conservation of momentum

$$\frac{\partial(\rho\mathbf{u})}{\partial t} + div(\rho\mathbf{u}\mathbf{u}) = -grad(p) + div(\mu\, grad(\mathbf{u})) + \rho\mathbf{f}$$

μ dynamic viscosity [kg/(m.s)]

$\mathbf{f}$ body force [m/s^2]

- Conservation of energy

$$\frac{\partial(\rho T)}{\partial t} + div(\rho T\mathbf{u}) = div(\Gamma\, grad(T)) + \frac{S}{c_p}$$

$\Gamma = k/c_p$

k thermal conductivity [W/(m.K)]

c_p specific heat capacity [J/(kg.K)]

S source [W/m^3]

Governing equations: compressible flows

- Unknowns $\mathbf{u} = (u, v, w),\ p,\ \rho,\ T$ functions of $\mathbf{x},\ t$

- Conservation of mass (continuity equation)

$$\frac{\partial \rho}{\partial t} + div(\rho \mathbf{u}) = 0$$

- Conservation of momentum

$$\frac{\partial (\rho \mathbf{u})}{\partial t} + div(\rho \mathbf{u}\mathbf{u}) = -grad(p) + grad\left(\frac{1}{3}\mu\, div(\mathbf{u})\right) + div(\mu\, grad(\mathbf{u})) + \rho \mathbf{f}$$

- Conservation of energy

$$\frac{\partial (\rho T)}{\partial t} + div(\rho T \mathbf{u}) = div(\Gamma\, grad(T)) + \frac{S}{c_p}$$

- Equation of state

$$p = p(\rho)$$

General conservation equation

- Unknown ϕ function of $\mathbf{x}, t$

$$\boxed{\frac{\partial(\rho\phi)}{\partial t}} + \boxed{div(\rho\phi\mathbf{u})} = \boxed{div(\Gamma\, grad(\phi))} + \boxed{S}$$

unsteadiness (rate of change) | convection (transport) | diffusion | creation/destruction (source/sink)

$(\phi = 1)$
$$\boxed{\frac{\partial\rho}{\partial t}} + \boxed{div(\rho\mathbf{u})} = 0$$

$(\phi = \mathbf{u})$
$$\boxed{\frac{\partial(\rho\mathbf{u})}{\partial t}} + \boxed{div(\rho\mathbf{u}\mathbf{u})} = \boxed{div\left[\left(-p - \frac{2}{3}\mu\, div(\mathbf{u})\right)\mathbf{I} + 2\mu\mathbf{d}\right]} + \boxed{\rho\mathbf{f}}$$

$(2\mathbf{d} = grad(\mathbf{u}) + grad(\mathbf{u})^T)$

$(\phi = T)$
$$\boxed{\frac{\partial(\rho T)}{\partial t}} + \boxed{div(\rho T\mathbf{u})} = \boxed{div(\Gamma\, grad(T))} + \boxed{\frac{S}{c_p}}$$

Simple "model" equations

- General conservation equation:

$$\boxed{\frac{\partial(\rho\phi)}{\partial t}} + \boxed{div(\rho\phi\mathbf{u})} = \boxed{div(\Gamma\, grad(\phi))} + \boxed{S}$$

unsteadiness convection diffusion source

- Steady/unsteady diffusion (e.g. heat conduction):

$$\boxed{div(\Gamma\, grad(\phi))} + S = 0$$

$$\boxed{\frac{\partial(\rho\phi)}{\partial t}} = div(\Gamma\, grad(\phi))$$

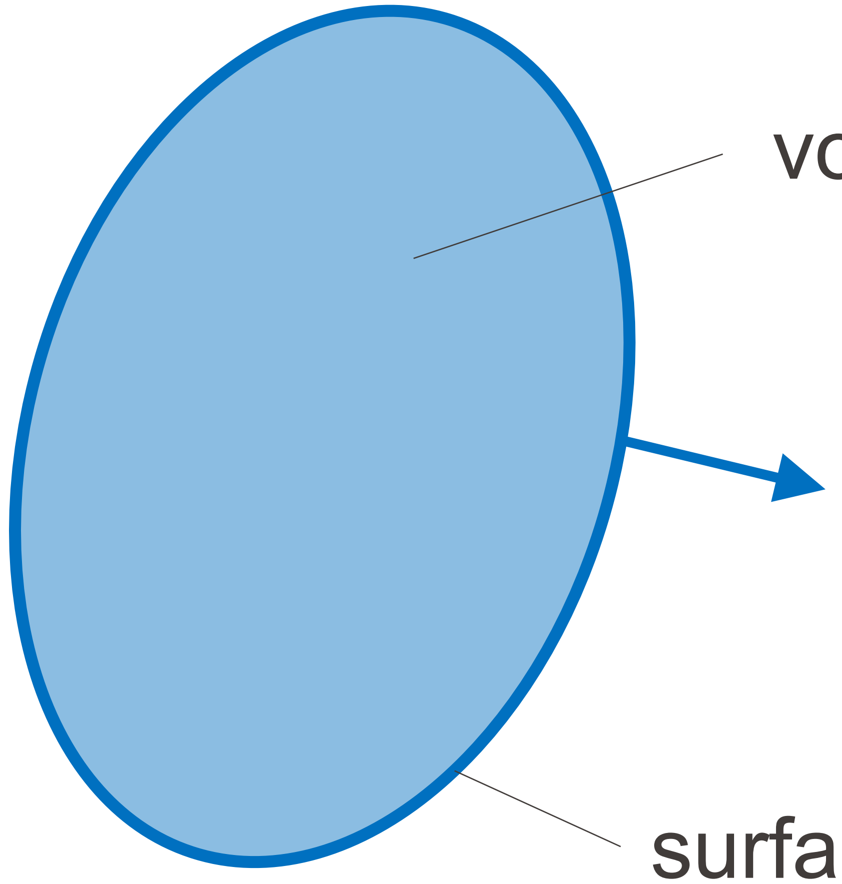
- Steady convection-diffusion (transport of a scalar, e.g. dye, salt, chemical species):

$$\boxed{div(\rho\phi\mathbf{u})} = div(\Gamma\, grad(\phi))$$

General conservation equation: integral form

- Unknown ϕ function of $\mathbf{x}, t$

$$\boxed{\frac{\partial(\rho\phi)}{\partial t}} + \boxed{div(\rho\phi\mathbf{u})} = \boxed{div(\Gamma\, grad(\phi))} + \boxed{S}$$



- Integrate over a control volume (CV)

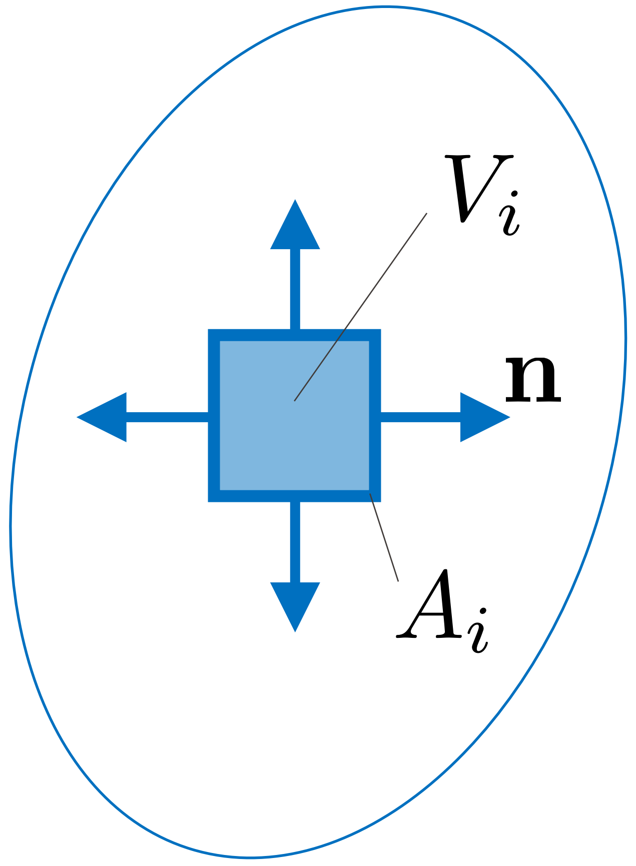
$$\frac{\partial}{\partial t}\int_V \rho\phi\, dV + \boxed{\int_V div(\rho\phi\mathbf{u})\, dV} = \boxed{\int_V div(\Gamma\, grad(\phi))\, dV} + \int_V S\, dV$$

- Use divergence theorem (Gauss's theorem) $\int_V div(\mathbf{a})\, dV = \oint_A \mathbf{a}\cdot\mathbf{n}\, dA$

$$\frac{\partial}{\partial t}\int_V \rho\phi\, dV + \boxed{\oint_A \rho\phi\mathbf{u}\cdot\mathbf{n}\, dA} = \boxed{\oint_A \Gamma\, grad(\phi)\cdot\mathbf{n}\, dA} + \int_V S\, dV$$

Convective and diffusive terms now expressed as fluxes through surface

Local/global conservation

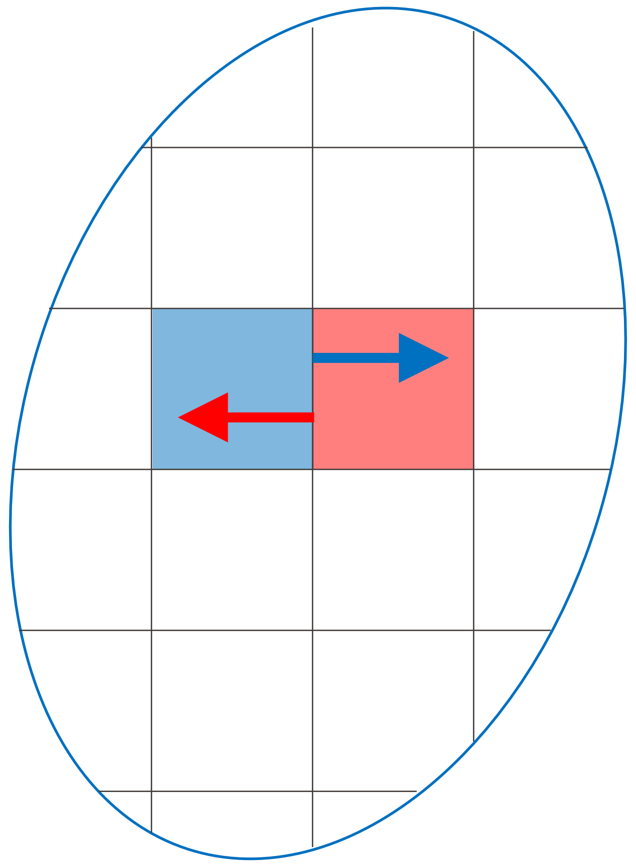


- Conservation equation holds for any CV, in particular any sub-domain V_i :

$$\frac{\partial}{\partial t}\int_{V_i} \rho\phi \, dV + \oint_{A_i} \rho\phi \mathbf{u}\cdot\mathbf{n}\, dA = \oint_{A_i} \Gamma\, grad(\phi)\cdot\mathbf{n}\, dA + \int_{V_i} S\, dV \quad (*)$$

- If divide the domain into a tiling of small CVs (no gap or overlap, each face shared by exactly 2 CVs), local conservation in each CV ensures global conservation:

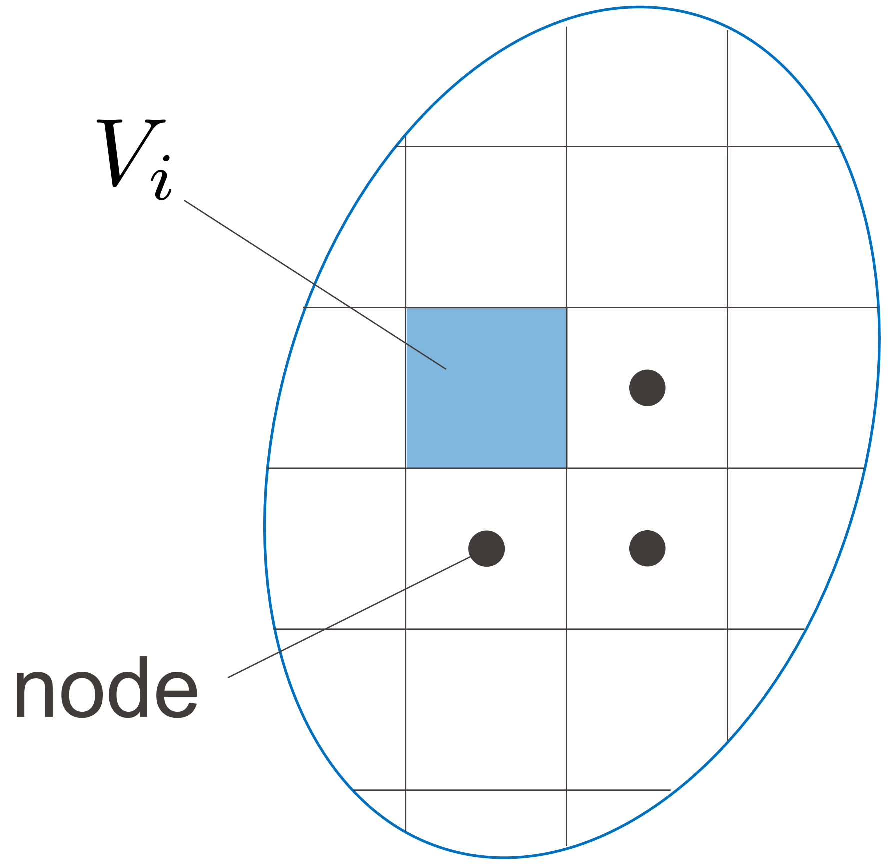
Fluxes through inner surfaces cancel out



$$\sum_i \int_{V_i} dV = \int_V dV \qquad \sum_i \oint_{A_i} \mathbf{n}\, dA = \int_A \mathbf{n}\, dA$$

$$\sum_i (*) = \frac{\partial}{\partial t}\int_{V} \rho\phi \, dV + \oint_{A} \rho\phi \mathbf{u}\cdot\mathbf{n}\, dA = \oint_{A} \Gamma\, grad(\phi)\cdot\mathbf{n}\, dA + \int_{V} S\, dV$$

The finite volume method



- Divide the domain into a tiling of CVs
- Assign a computational node to each CV
- Discretize the unknown with nodal values:

$$\phi(\mathbf{x}) \to \boldsymbol{\phi} = (\phi_1, \phi_2 \ldots, \phi_i \ldots)$$

- Approximate each term of the governing equation as a function of the nodal values (see later)
- (Specify boundary conditions; discretize the time derivative; see later)

- These steps transform the original PDE into a system of algebraic equations:

$$\frac{\partial}{\partial t}\int_V \rho\phi\, dV + \oint_A \rho\phi\mathbf{u}\cdot\mathbf{n}\, dA = \oint_A \Gamma\, grad(\phi)\cdot\mathbf{n}\, dA + \int_V S\, dV \quad \to \boxed{\mathbf{A}(\boldsymbol{\phi}) = \mathbf{0}}$$

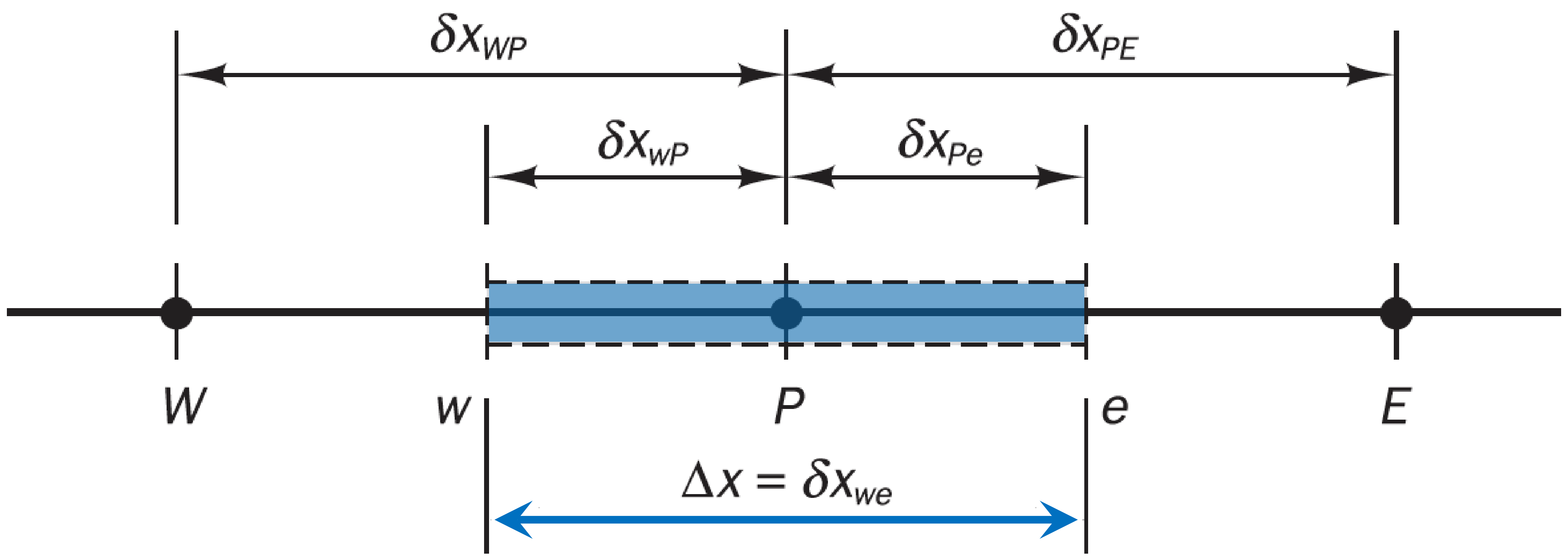
The finite volume method

- Notations (structured grids):
    - Nodal point *P*, neighbors *W*, *E*…
    - Faces: *w*, *e*…
    - CV dimensions: $\Delta x,\ \Delta y \ldots$
    - Node-node distances: $\delta x_{WP},\ \delta x_{PE} \ldots$
    - Node-face distances: $\delta x_{wP},\ \delta x_{Pe} \ldots$
    - Nodal values (the discretized solution):

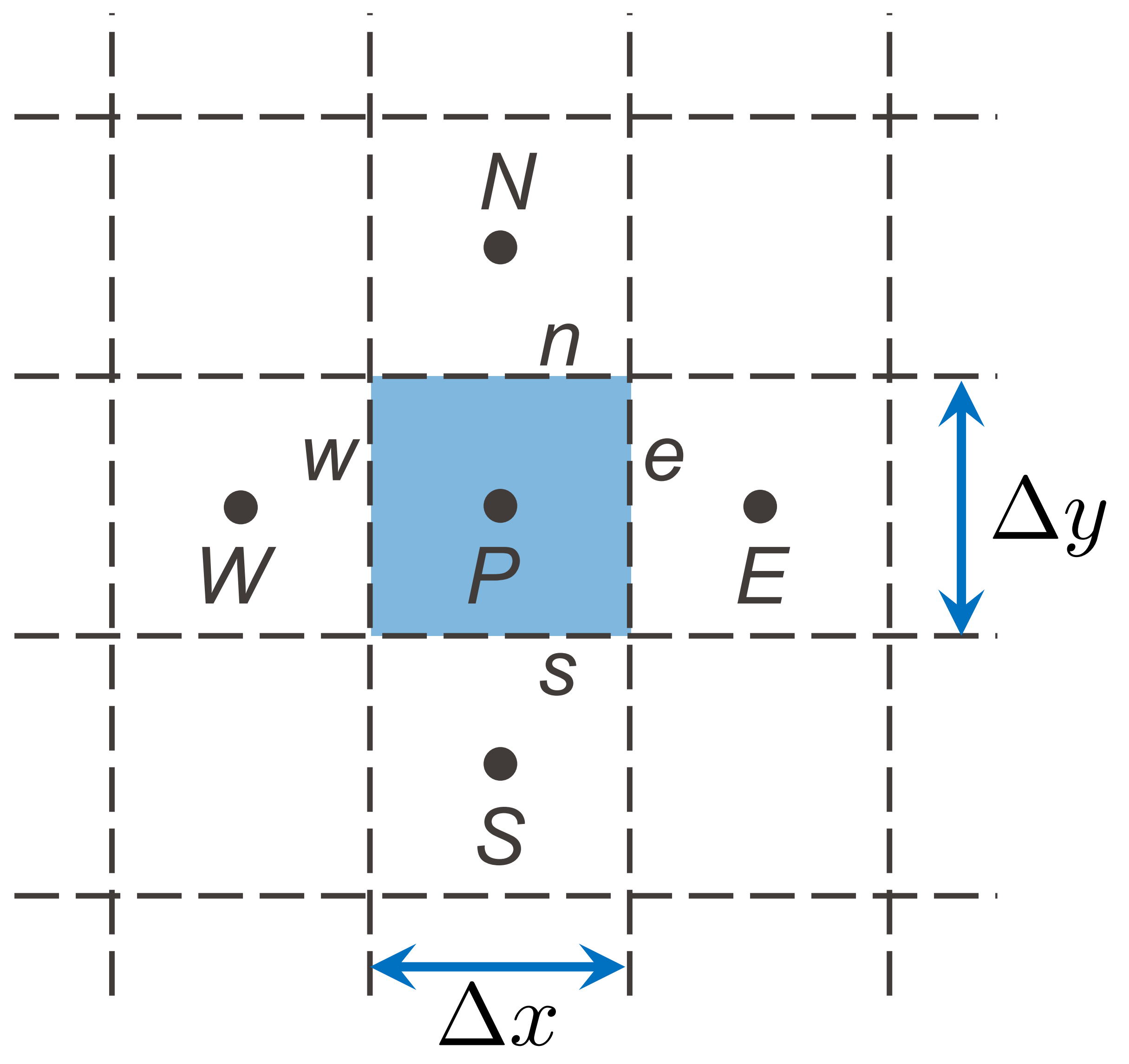
$$\phi_P, \phi_W, \phi_E \ldots$$

    - Face values (not known; to be interpolated from nodal values): $\phi_w, \phi_e \ldots$

1D example:



2D example:



The FVM: volume integrals

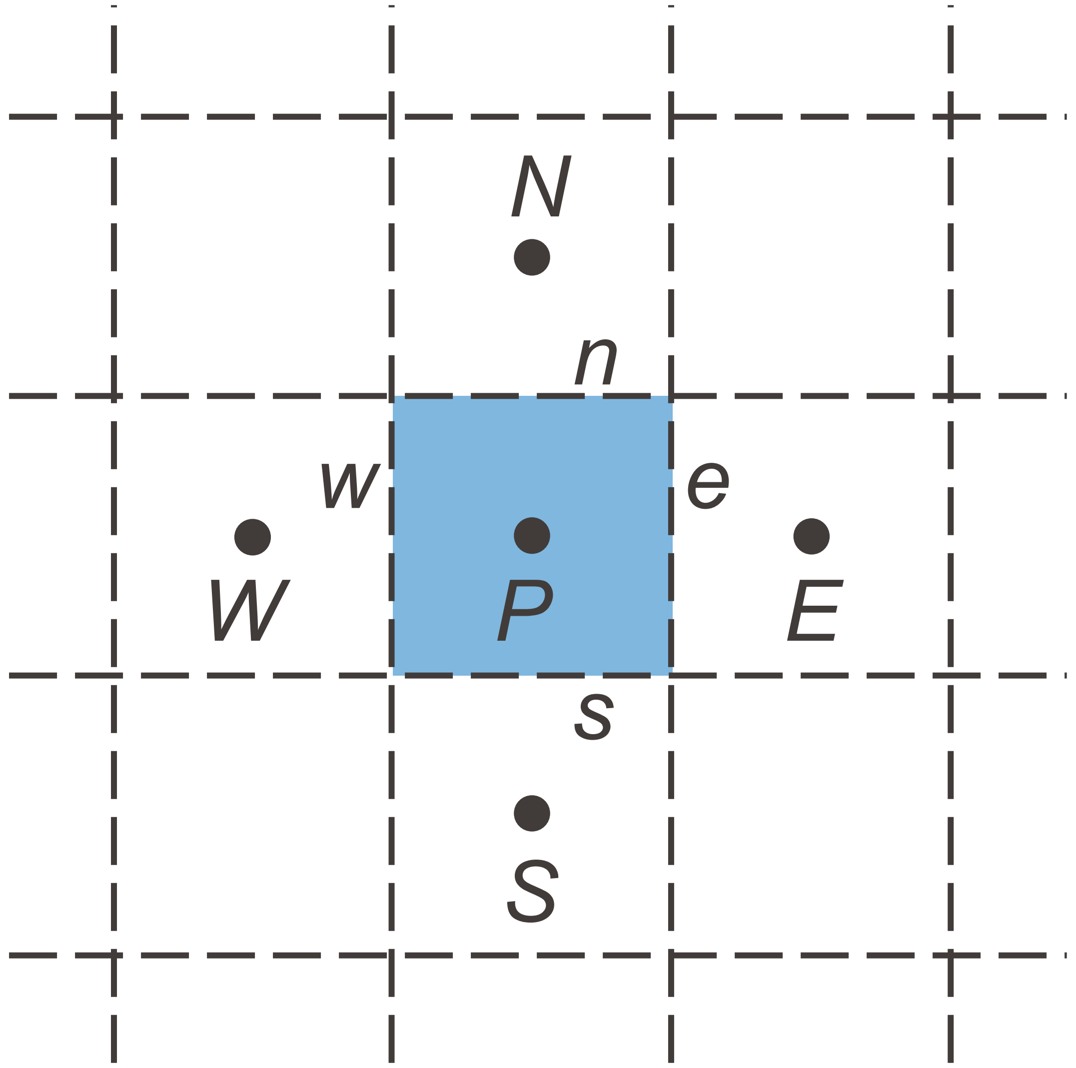
- Local conservation equation:

$$\boxed{\frac{\partial}{\partial t}\int_{V_i}\rho\phi\,dV} + \oint_{A_i}\rho\phi\mathbf{u}\cdot\mathbf{n}\,dA = \oint_{A_i}\Gamma\,grad(\phi)\cdot\mathbf{n}\,dA + \boxed{\int_{V_i}S\,dV}$$

- Approximation of volume integrals:

Nodal value * volume:

$$\int_{V_i}S\,dV = \overline{S}\,V_i \approx S_P\,V_i$$



Exact if S constant or linear;
2^{nd}-order approximation otherwise.

The FVM: surface integrals

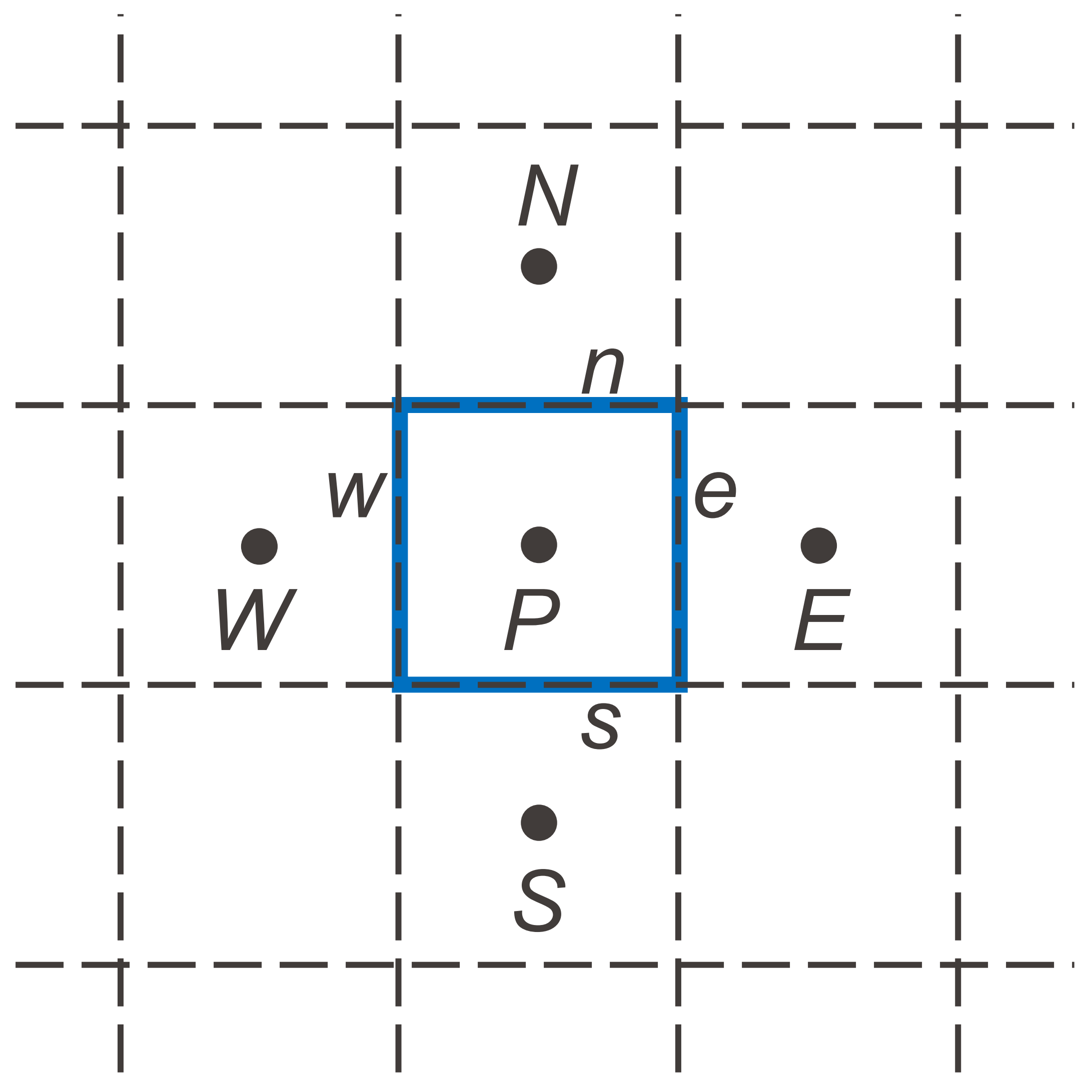
- Local conservation equation:

$$\frac{\partial}{\partial t}\int_{V_i} \rho\phi \, dV + \boxed{\oint_{A_i} \rho\phi \mathbf{u}\cdot\mathbf{n}\, dA} = \boxed{\oint_{A_i} \Gamma \, grad(\phi)\cdot\mathbf{n}\, dA} + \int_{V_i} S\, dV$$

- Approximation of surface integrals:

For instance, flux through face *e*: $\int_e f\, dA$

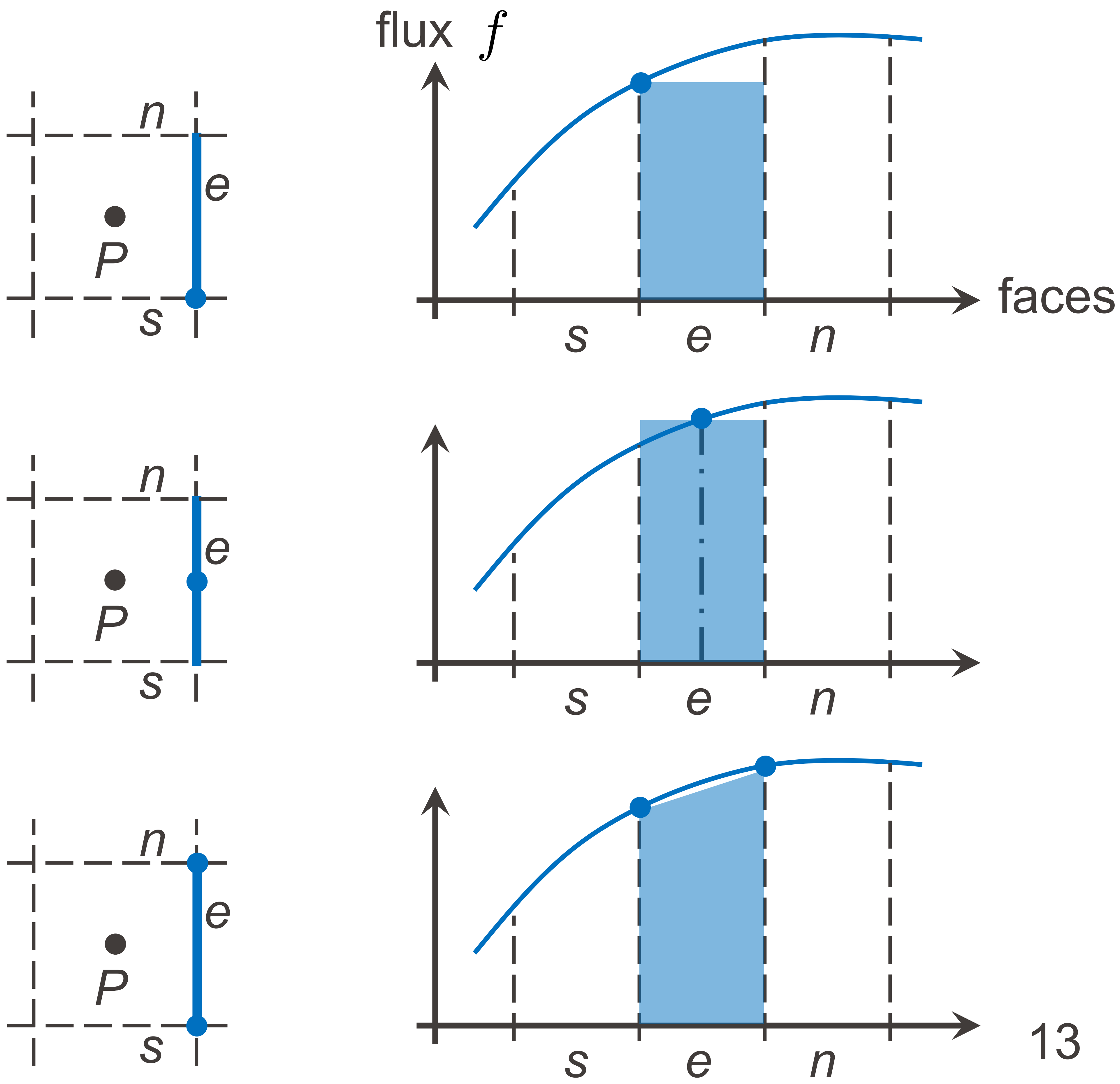
1. Approximate the integral from discrete value(s) on the face,
2. Then interpolate face value(s) from nodal values.



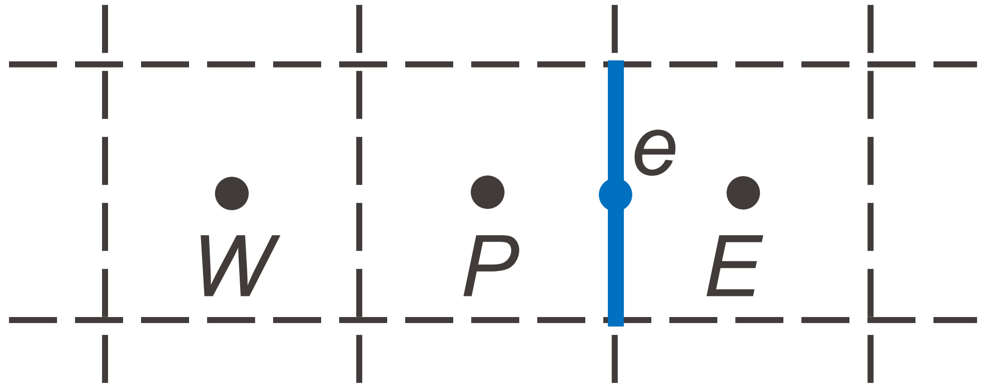
The FVM: surface integrals

1. Approximate the integral from discrete value(s) on the face:

   - Left or right Riemann sum: only 1st-order accurate
   $$\int_e f\,dA \approx f_{se} A_e \quad \text{or} \quad f_{ne} A_e$$
   - Midpoint rule (use value at face center): 2nd-order accurate
   $$\int_e f\,dA \approx f_e A_e$$
   - Trapezoidal rule: 2nd-order too, but more complicated
   $$\int_e f\,dA \approx \frac{f_{ne} + f_{se}}{2} A_e$$



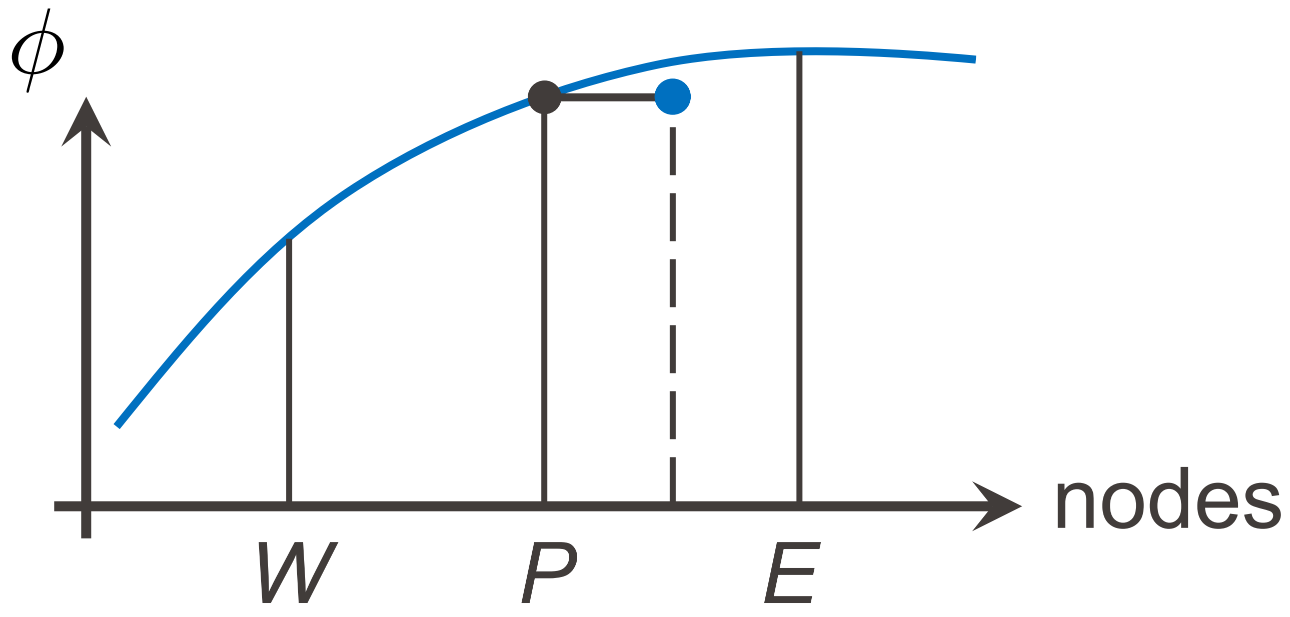
The FVM: surface integrals



2. Interpolate face value from nodal values: deduce schemes from Taylor expansions such as

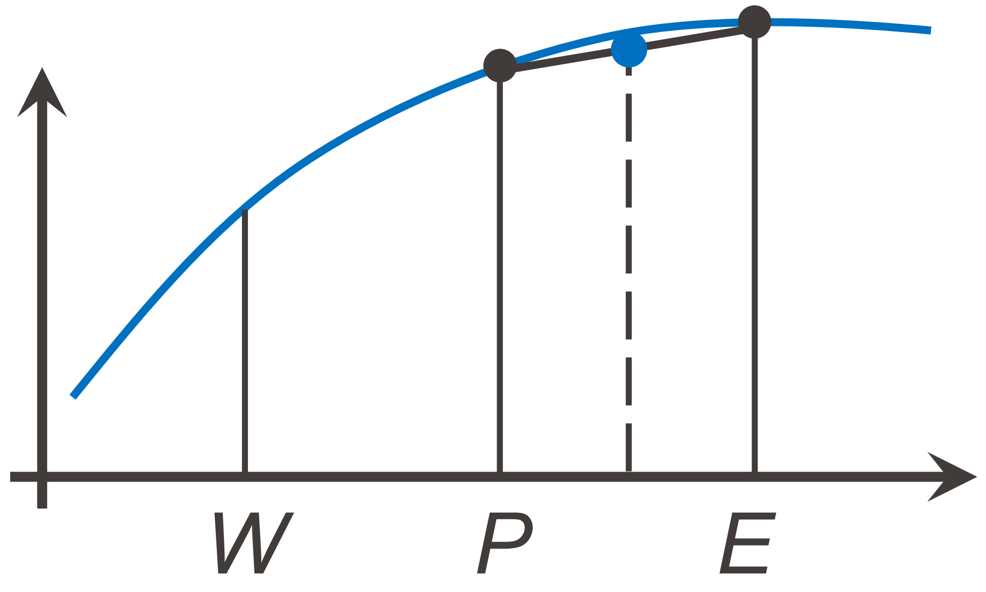
$$\phi_E = \phi_e + (x_E - x_e)\left(\frac{\partial \phi}{\partial x}\right)_e + \frac{(x_E - x_e)^2}{2}\left(\frac{\partial^2 \phi}{\partial x^2}\right)_e + \dots$$

$$\phi_P = \phi_e + (x_P - x_e)\left(\frac{\partial \phi}{\partial x}\right)_e + \frac{(x_P - x_e)^2}{2}\left(\frac{\partial^2 \phi}{\partial x^2}\right)_e + \dots$$



- Constant interpolation (UD, “upwind differencing”), 1st-order accurate

$$\phi_e \approx \phi_P \quad \text{or} \quad \phi_E$$

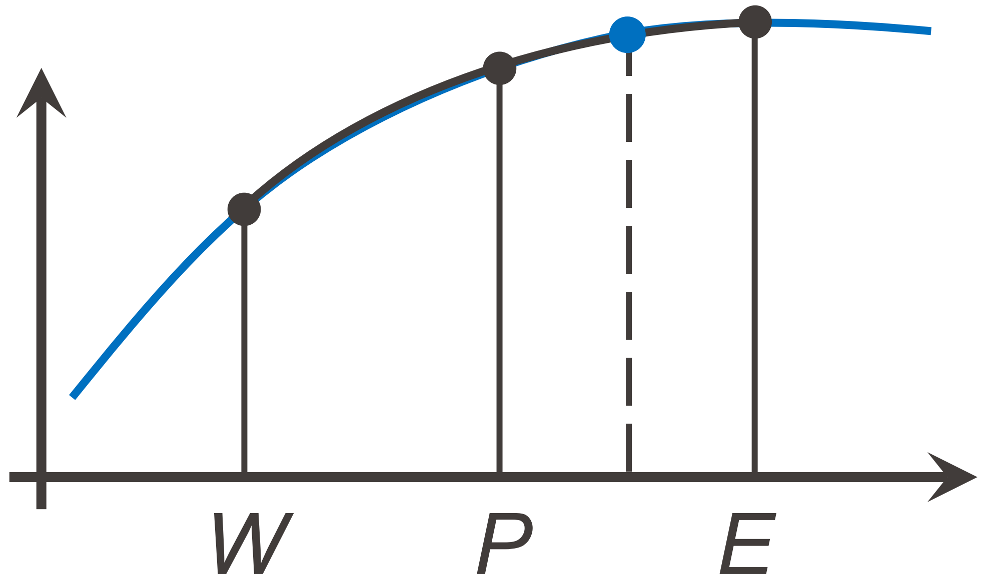


- Linear interpolation (CD, “central differencing”), 2nd-order

$$\phi_e \approx \frac{\phi_P + \phi_E}{2} \; (*) \qquad \text{and} \qquad \left(\frac{\partial \phi}{\partial x}\right)_e \approx \frac{\phi_E - \phi_P}{\delta x_{PE}}$$

- Quadratic interpolation (QUICK, “quadratic upwind interp. for convective kinematics”), 3rd-order

$$\phi_e \approx \frac{-\phi_W + 6\phi_P + 3\phi_E}{8} \; (*)$$



($*$ on a uniform grid)

Simple "model" equations

- General conservation equation:

$$\boxed{\frac{\partial(\rho\phi)}{\partial t}} + \boxed{div(\rho\phi\mathbf{u})} = \boxed{div(\Gamma\, grad(\phi))} + \boxed{S}$$

unsteadiness convection diffusion source

- Steady/unsteady diffusion (e.g. heat conduction):

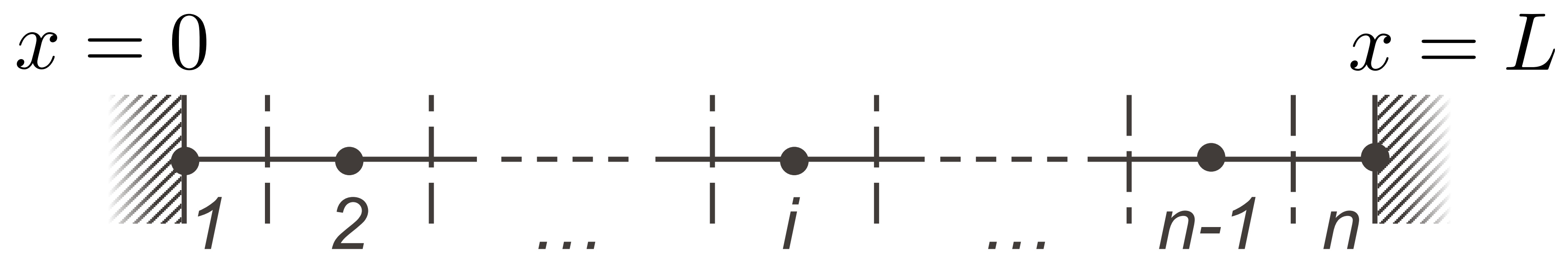
$$\boxed{div(\Gamma\, grad(\phi))} + S = 0$$

$$\boxed{\frac{\partial(\rho\phi)}{\partial t}} = div(\Gamma\, grad(\phi))$$

- Steady convection-diffusion (transport of a scalar, e.g. dye, salt, chemical species):

$$\boxed{div(\rho\phi\mathbf{u})} = div(\Gamma\, grad(\phi))$$

Steady diffusion: 1D heat conduction

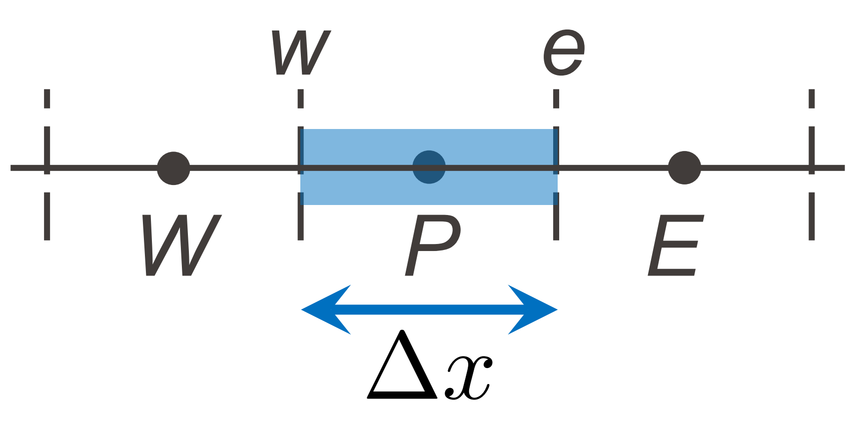


- Problem:
  - Equation: $\dfrac{\partial}{\partial x}\left(k\dfrac{\partial T}{\partial x}\right) + S = 0$
  - Domain: $x \in [0, L]$
  - Known thermal conductivity and source term: $k(x), S(x)$
  - Boundary conditions: $T(0) = T_a, \quad T(L) = T_b$
  - Example: solid or fluid cylinder, constant cross-section area A

- Discretize into *n* CVs, *n* nodes: $T(\mathbf{x}) \rightarrow \boldsymbol{T} = (T_1, T_2 \ldots, T_i \ldots, T_n)$
- Integration over CV + divergence theorem:

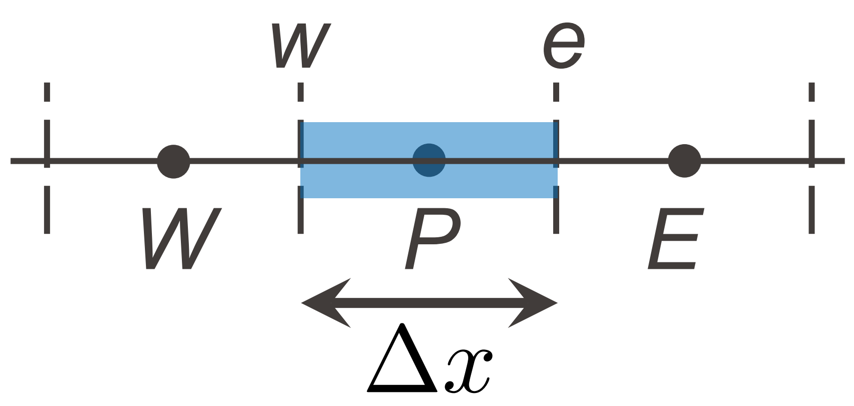
$$0 = \int_{x_w}^{x_e} \left(\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right) + S \right) dx = \left[k\frac{\partial T}{\partial x} \right]_{x_w}^{x_e} + \int_{x_w}^{x_e} S\, dx$$

$$= \left(k\frac{\partial T}{\partial x}\right)_e - \left(k\frac{\partial T}{\partial x}\right)_w + \int_{x_w}^{x_e} S\, dx$$



Numerical Flow Simulation

Steady diffusion: 1D heat conduction



- Approximate surface integral (face fluxes) with CD, and volume integral (nodal value * volume):

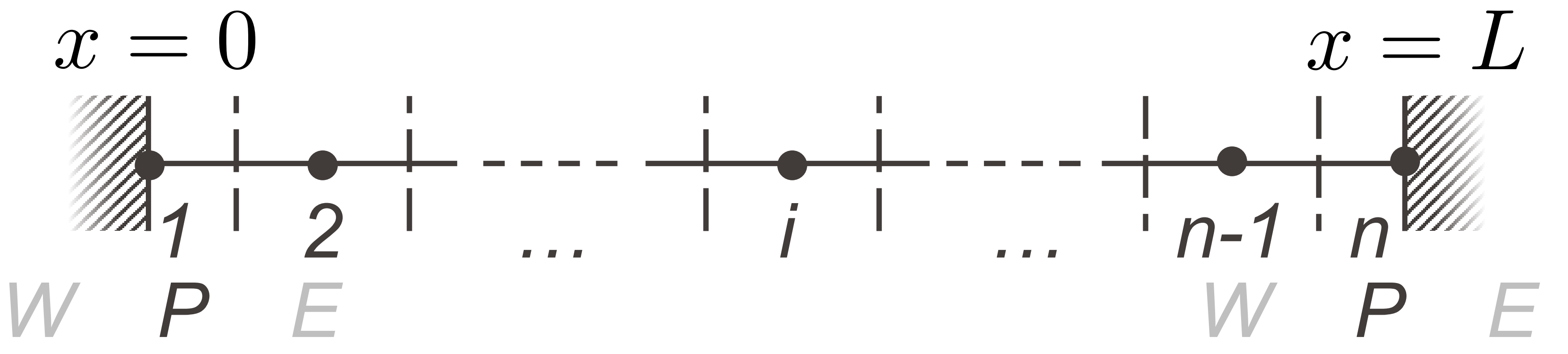
$$0 = \left(k\frac{\partial T}{\partial x}\right)_e - \left(k\frac{\partial T}{\partial x}\right)_w + \int_{x_w}^{x_e} S\,dx \approx k_e\frac{T_E - T_P}{\delta x_{PE}} - k_w\frac{T_P - T_W}{\delta x_{WP}} + S_P\Delta x$$

- Linear algebraic equation:

$$\left(\frac{k_e}{\delta x_{PE}} + \frac{k_w}{\delta x_{WP}}\right)T_P = \left(\frac{k_w}{\delta x_{WP}}\right)T_W + \left(\frac{k_e}{\delta x_{PE}}\right)T_E + S_P\Delta x$$

$$a_P T_P = a_W T_W + a_E T_E + b$$

- Boundary conditions:



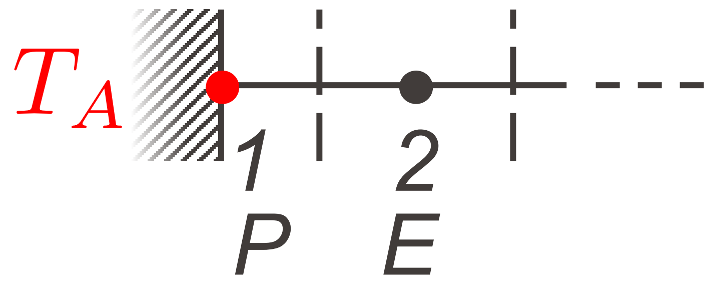
$$T(0) = T_1 = T_A \quad \rightarrow 1.T_P = 0.T_W + 0.T_E + T_A$$

$$T(L) = T_n = T_B \quad \rightarrow 1.T_P = 0.T_W + 0.T_E + T_B$$

Other possible boundary conditions

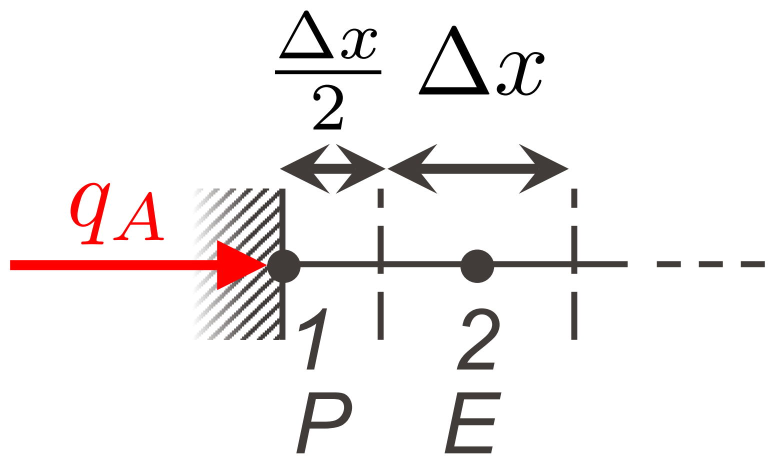
(in red below: known quantities)

- **Dirichlet**: known value of the solution ϕ_1, for ex. $T_1 = T_A$



$$\rightarrow \ 1.T_P = 0.T_E + T_A$$

- **Neumann**: known derivative $\left(\frac{\partial \phi}{\partial \mathbf{n}}\right)_1$, for ex. $k_1 \left(\frac{\partial T}{\partial x}\right)_1 = -q_A$



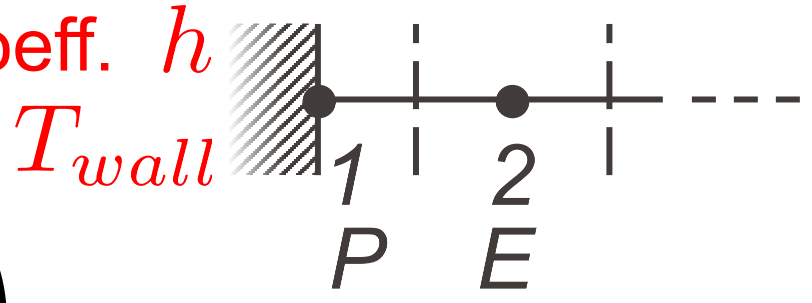
$$0 = \left(k\frac{\partial T}{\partial x}\right)_e - \left(k\frac{\partial T}{\partial x}\right)_w + \int_{x_w}^{x_e} S\,dx \ \approx k_e \frac{T_E - T_P}{\delta x_{PE}} + q_A + S_P \frac{\Delta x}{2}$$

$$\rightarrow \left(\frac{k_e}{\delta x_{PE}}\right) T_P = \left(\frac{k_e}{\delta x_{PE}}\right) T_E + \left(q_A + S_P \frac{\Delta x}{2}\right)$$

- **Robin**: relationship between solution and derivative $a\phi_1 + b\left(\frac{\partial \phi}{\partial \mathbf{n}}\right)_1 = c$

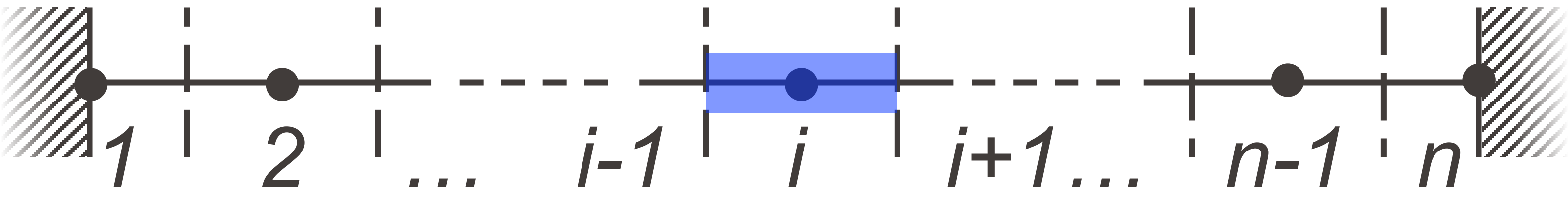
for ex. $-k_1 \left(\frac{\partial T}{\partial x}\right)_1 = h(T_{wall} - T_1)$

Heat transfer coeff. h



$$\rightarrow \left(\frac{k_e}{\delta x_{PE}} + h\right) T_P = \left(\frac{k_e}{\delta x_{PE}}\right) T_E + \left(hT_{wall} + S_P \frac{\Delta x}{2}\right)$$

Assembling the algebraic system



- 1^{st} CV: $a_P T_P = a_E T_E + b \quad \rightarrow \quad a_{1,1}T_1 + a_{1,2}T_2 = b_1$
- i^{th} CV: $a_P T_P = a_W T_W + a_E T_E + b \quad \rightarrow \quad a_{i,i-1}T_{i-1} + a_{i,i}T_i + a_{i,i+1}T_{i+1} = b_i$
- n^{th} CV: $a_P T_P = a_W T_W + b \quad \rightarrow \quad a_{n,n}T_n + a_{n,n-1}T_{n-1} = b_n$

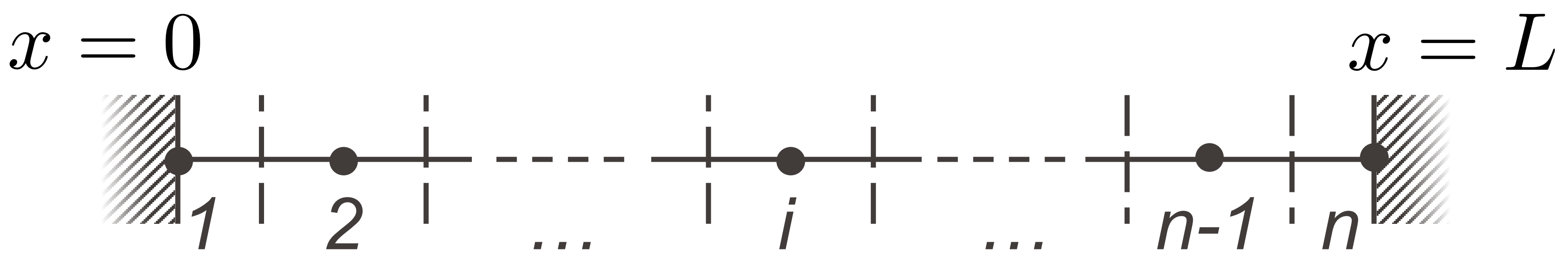
$a_{i,j}$: i^{th} equation (conservation in i^{th} CV), contribution from j^{th} CV

$$\begin{bmatrix} a_{1,1} & a_{1,2} & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ a_{2,1} & a_{2,2} & a_{2,3} & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & a_{3,2} & a_{3,3} & a_{3,4} & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & a_{i,i-1} & a_{i,i} & a_{i,i+1} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 & 0 & a_{n-2,n-3} & a_{n-2,n-2} & a_{n-2,n-1} & 0 \\ 0 & \cdots & 0 & 0 & 0 & 0 & a_{n-1,n-2} & a_{n-1,n-1} & a_{n-1,n} \\ 0 & \cdots & 0 & 0 & 0 & 0 & 0 & a_{n,n-1} & a_{n,n} \end{bmatrix} \begin{pmatrix} T_1 \\ T_2 \\ T_3 \\ \vdots \\ T_i \\ \vdots \\ T_{n-2} \\ T_{n-1} \\ T_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_i \\ \vdots \\ b_{n-2} \\ b_{n-1} \\ b_n \end{pmatrix}$$

Linear system: $\mathbf{AT} = \mathbf{b}$

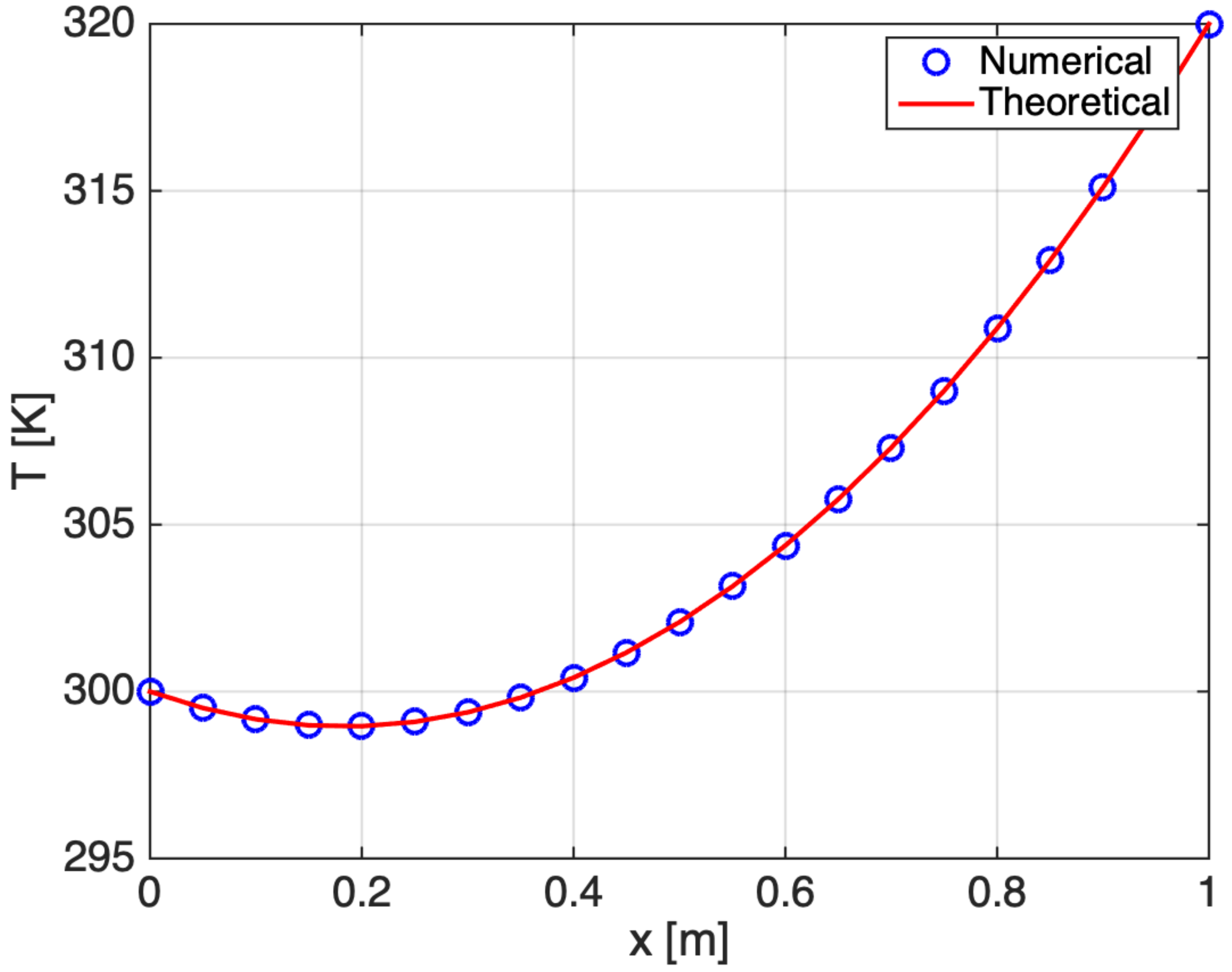
$\mathbf{A}$: *n* x *n* matrix (here tri-diagonal), $\mathbf{T}, \mathbf{b}$: *n* x 1 vectors $\quad \rightarrow \mathbf{T} = \mathbf{A}^{-1}\mathbf{b}$

Demo: 1D diffusion with Matlab



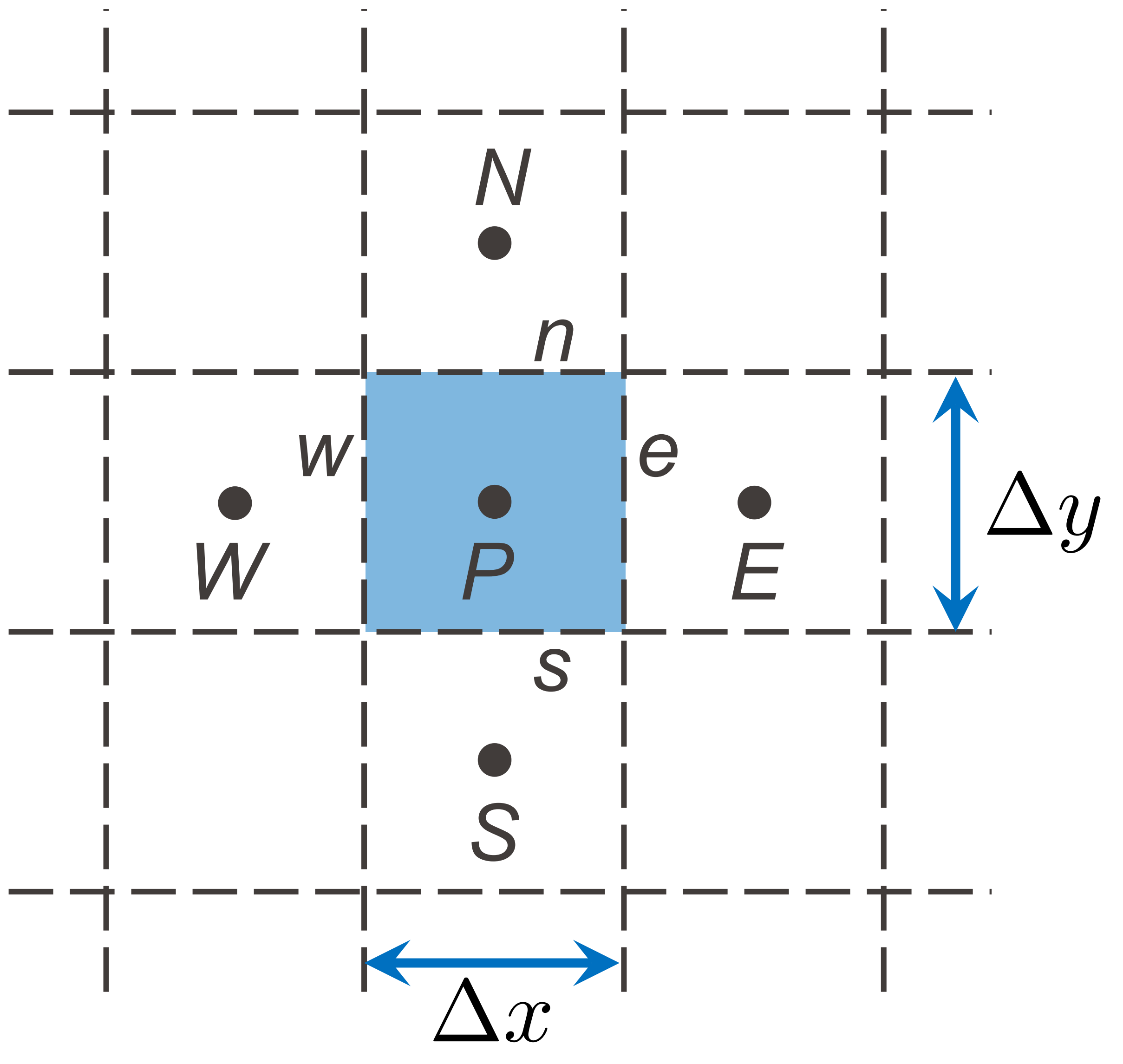

```
%%%%%%%%%%%%%% Physical parameters
L = 1;
Ta = 300;
Tb = 320;
k = 400;
Sc = 5000;    Sl = -100;

%%%%%%%%%%%%%% Numerical parameters
n = 21;

%%%%%%%%%%%%%%% Grid generation
x0 = linspace(0,L,n);

dx = L/(n-1);
Dx = dx;
```

```
%%%%%%%%%%%%%%% Create the matrix and right-hand side vector
A = zeros(n,n);
b = zeros(n,1);

for i=2:n-1
    A(i,i-1) = -k/dx;
    A(i,i+1) = -k/dx;
    A(i,i)   = 2*k/dx -Sl*Dx;
    b(i)     = Sc*Dx;
end

% Impose boundary conditions
A(1,1) = 1;
b(1)   = Ta;
A(n,n) = 1;
b(n)   = Tb;

%%%%%%%%%%%%%%% Numerical solution
%T=inv(A)*b;
T = A\b;
```



Numerical Flow Simulation

2D steady diffusion



$$div(k\,grad(T)) + S = 0$$

$$\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right) + S = 0$$

- Integration over CV + divergence theorem

$$0 = \int_V \left(div(k\,grad(T)) + S\right)\,dV = \oint_A k\,grad(T)\cdot\mathbf{n}\,dA + \int_V S\,dV$$

$$= \int_e k\frac{\partial T}{\partial x}dy - \int_w k\frac{\partial T}{\partial x}dy + \int_n k\frac{\partial T}{\partial y}dx - \int_s k\frac{\partial T}{\partial y}dx + \int_{y_s}^{y_n}\int_{x_w}^{x_e} S\,dx\,dy$$

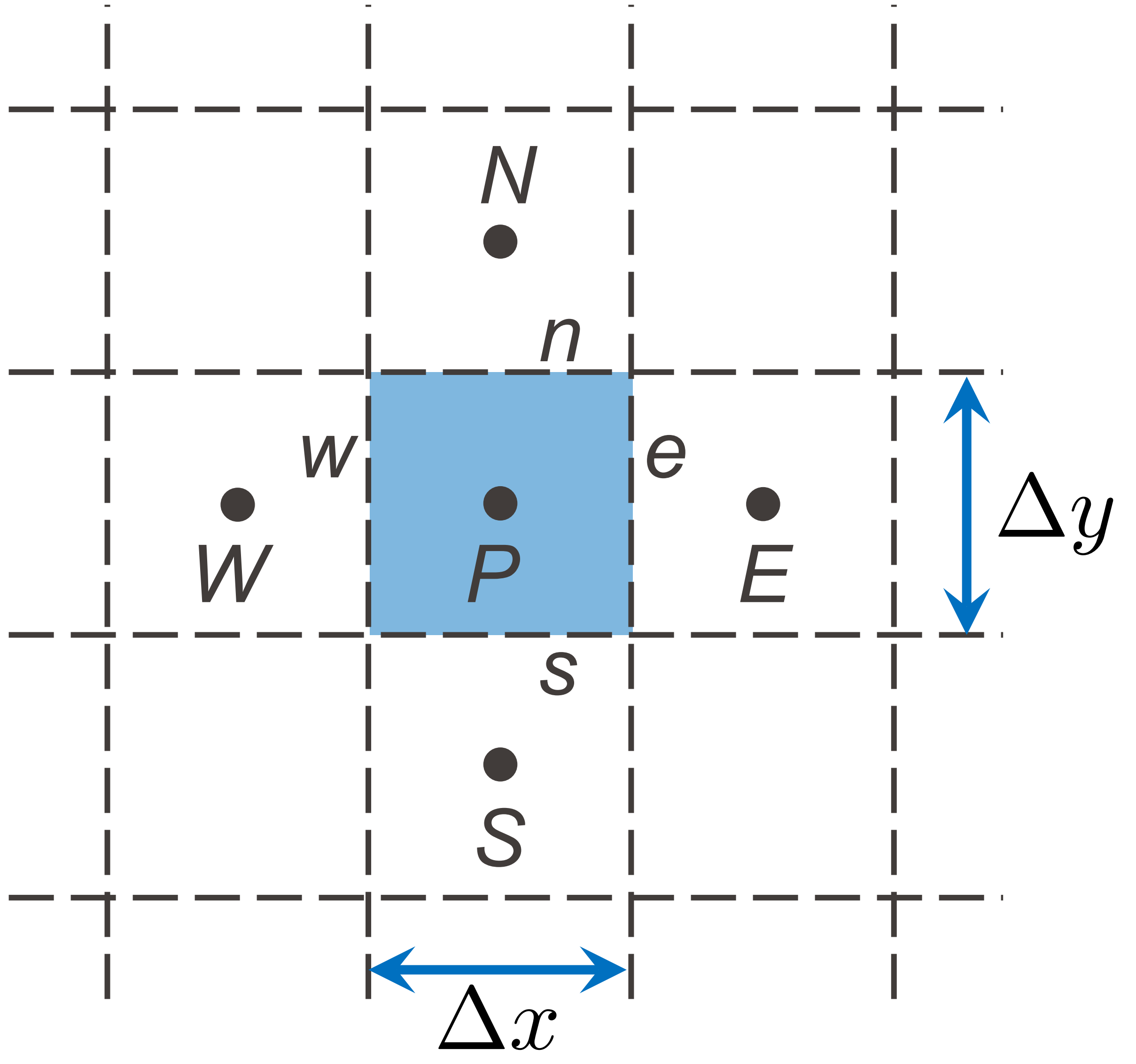
- If rectangular grid: direct integration also possible

$$0 = \int_{y_s}^{y_n}\int_{x_w}^{x_e}\left(\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right) + S\right)\,dx\,dy$$

$$= \int_{y_s}^{y_n}\left[k\frac{\partial T}{\partial x}\right]_{x_w}^{x_e} dy + \int_{x_w}^{x_e}\left[k\frac{\partial T}{\partial y}\right]_{y_s}^{y_n} dx + \int_{y_s}^{y_n}\int_{x_w}^{x_e} S\,dx\,dy$$

$$= \int_{y_s}^{y_n}\left[\left(k\frac{\partial T}{\partial x}\right)_e - \left(k\frac{\partial T}{\partial x}\right)_w\right]dy + \int_{x_w}^{x_e}\left[\left(k\frac{\partial T}{\partial y}\right)_n - \left(k\frac{\partial T}{\partial y}\right)_s\right]dx + \int_{y_s}^{y_n}\int_{x_w}^{x_e} S\,dx\,dy$$

2D steady diffusion



$$0 = \int_e k\frac{\partial T}{\partial x}dy - \int_w k\frac{\partial T}{\partial x}dy + \int_n k\frac{\partial T}{\partial y}dx - \int_s k\frac{\partial T}{\partial y}dx + \int_{y_s}^{y_n}\int_{x_w}^{x_e} S\,dx\,dy$$

- Approximation as function of nodal values (CD):

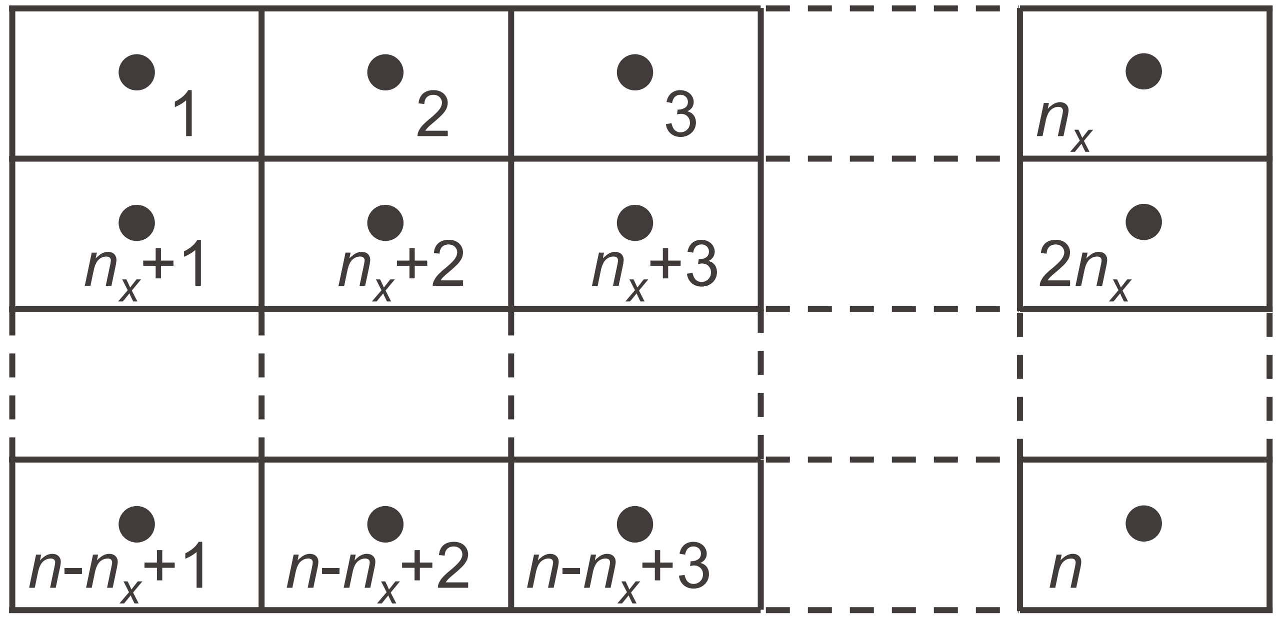
$$0 \approx \left[k_e\frac{T_E - T_P}{\delta x_{PE}} - k_w\frac{T_P - T_W}{\delta x_{WP}}\right]\Delta y + \left[k_n\frac{T_N - T_P}{\delta x_{PN}} - k_s\frac{T_P - T_S}{\delta x_{SP}}\right]\Delta x + S_P\Delta x\Delta y$$

- Linear algebraic equation:

$$\left(\frac{k_n\Delta x}{\delta x_{PN}} + \frac{k_s\Delta x}{\delta x_{SP}} + \frac{k_e\Delta y}{\delta x_{PE}} + \frac{k_w\Delta y}{\delta x_{WP}}\right)T_P =$$
$$\left(\frac{k_w\Delta y}{\delta x_{WP}}\right)T_W + \left(\frac{k_e\Delta y}{\delta x_{PE}}\right)T_E + \left(\frac{k_s\Delta x}{\delta x_{SP}}\right)T_S + \left(\frac{k_n\Delta x}{\delta x_{PN}}\right)T_N + S_P\Delta x\Delta y$$

$$a_P T_P = a_W T_W + a_E T_E + a_S T_S + a_N T_N + b = \sum_{nb} a_{nb}T_{nb} + b$$

2D steady diffusion

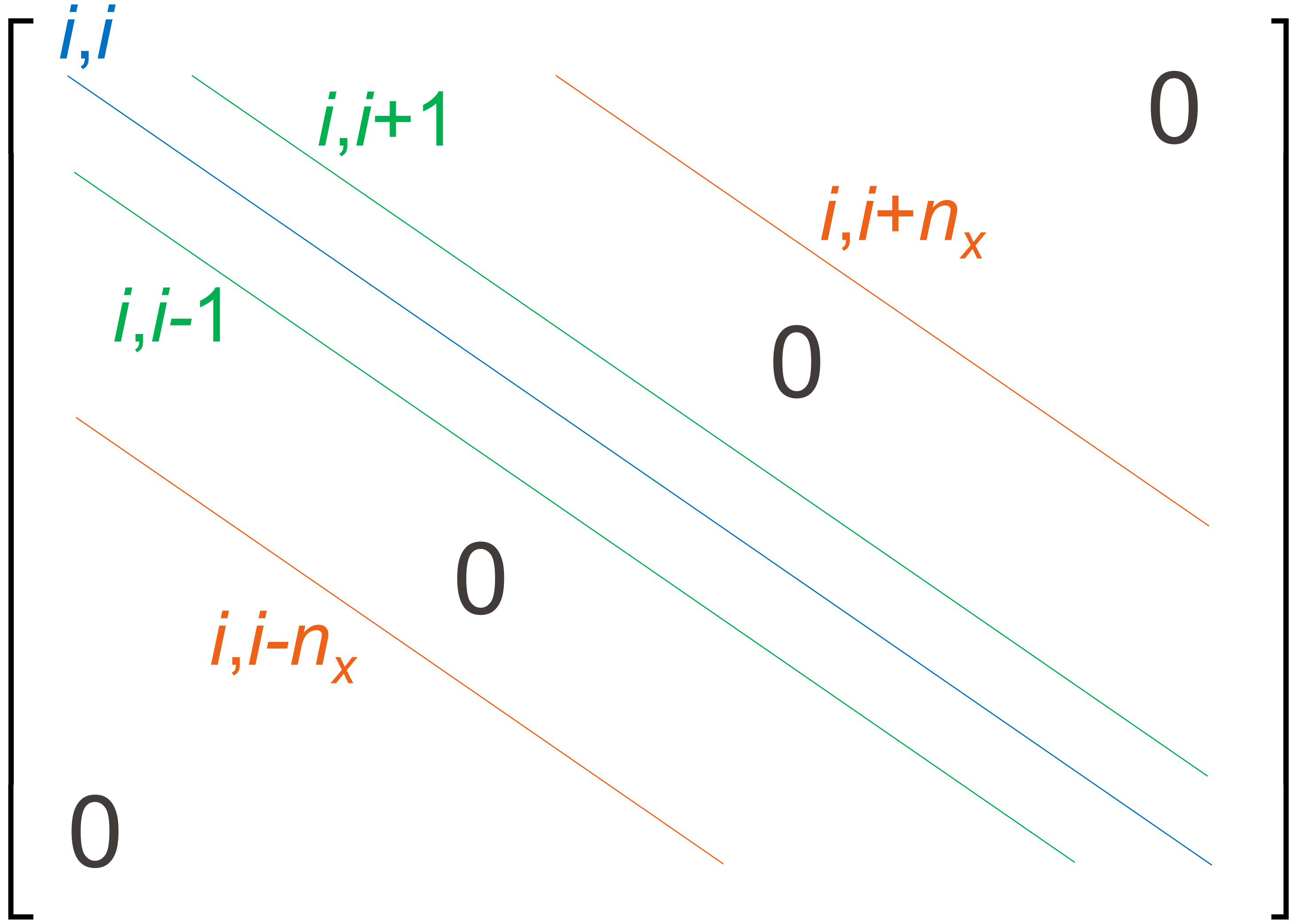


- On a structured quadrilateral mesh of $n=n_x \times n_y$ CVs and nodes: neighbor connectivity is known (no need to store info in memory)

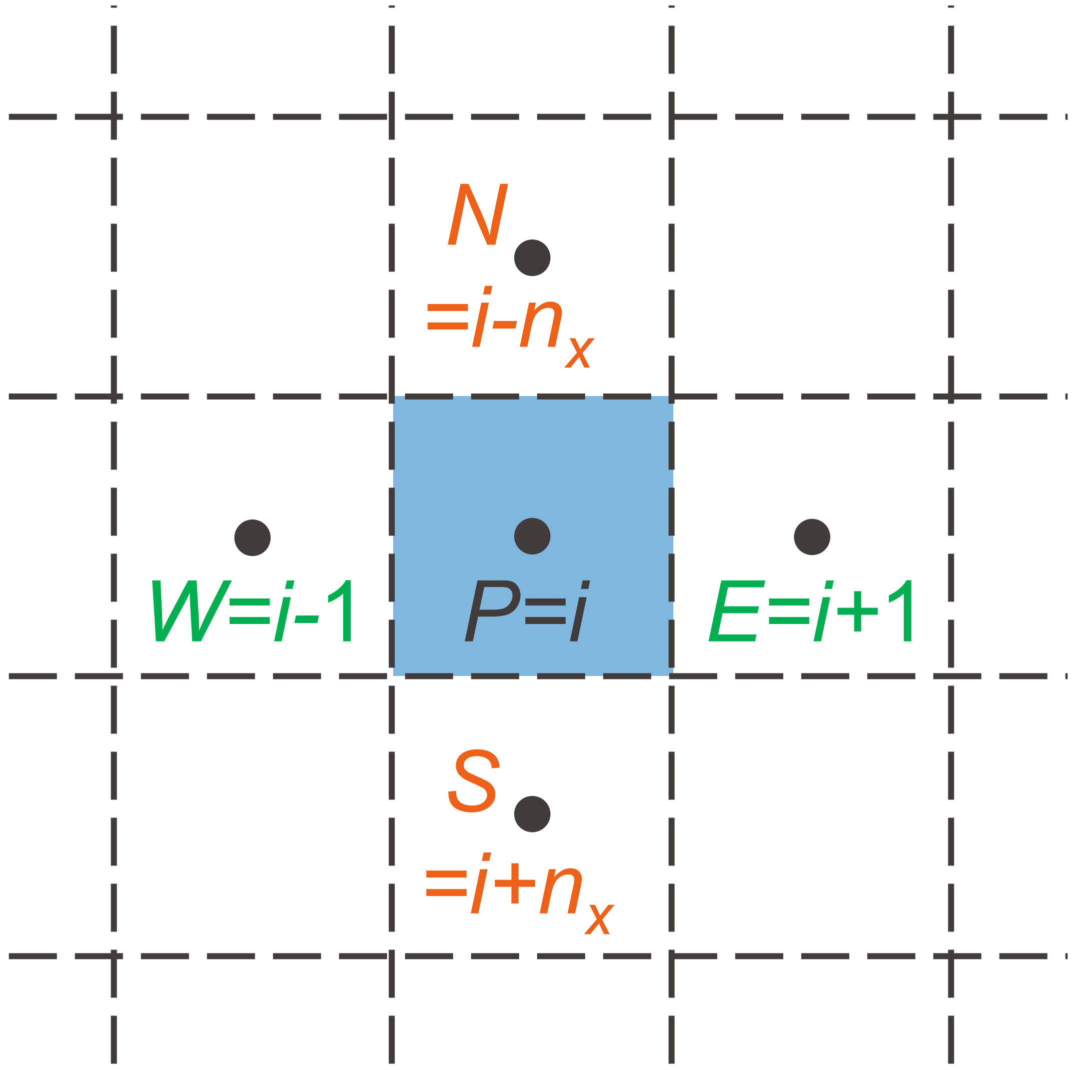
  → in the i^{th} CV, $a_P T_P = a_W T_W + a_E T_E + a_S T_S + a_N T_N + b$

  becomes $a_{i,i-n_x} T_{i-n_x} + a_{i,i-1} T_{i-1} + a_{i,i} T_i + a_{i,i+1} T_{i+1} + a_{i,i+n_x} T_{i+n_x} = b_i$

- Linear system: $\mathbf{A}\mathbf{T} = \mathbf{b}$, $\mathbf{A}$ sparse with 5 non-zero diagonals



$$\begin{pmatrix} T_1 \\ T_2 \\ T_3 \\ \vdots \\ T_i \\ \vdots \\ T_{n-2} \\ T_{n-1} \\ T_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_i \\ \vdots \\ b_{n-2} \\ b_{n-1} \\ b_n \end{pmatrix}$$



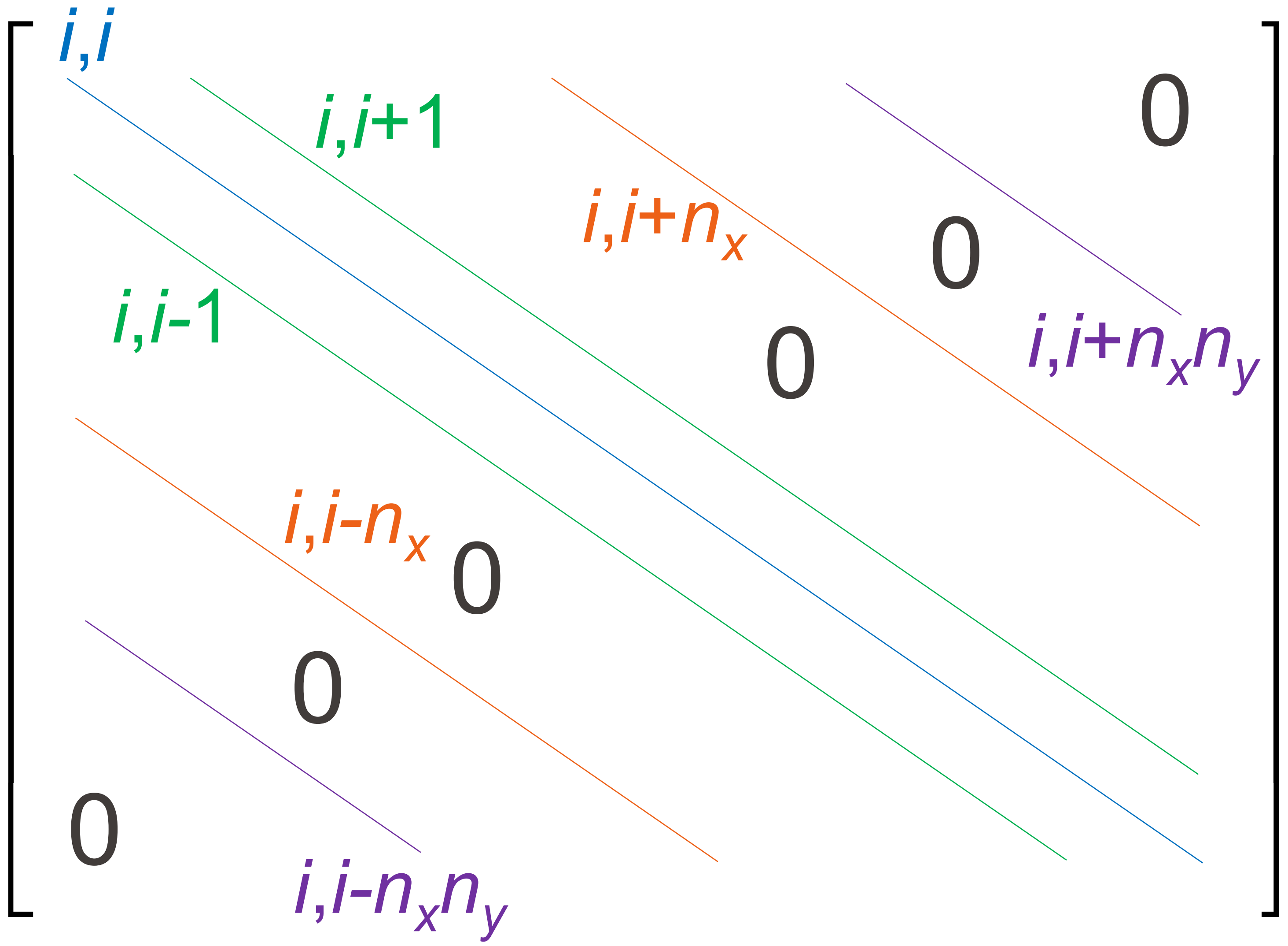
3D steady diffusion

- On a structured quadrilateral mesh of $n=n_x \times n_y \times n_z$ CVs and nodes: neighbor connectivity is known

  → in the i^{th} CV, $a_P T_P = a_W T_W + a_E T_E + a_S T_S + a_N T_N + a_T T_T + a_B T_B + b$

$$a_{i,i-n_x n_y} T_{i-n_x n_y} + a_{i,i-n_x} T_{i-n_x} + a_{i,i-1} T_{i-1} + a_{i,i} T_i + a_{i,i+1} T_{i+1} + a_{i,i+n_x} T_{i+n_x} + a_{i,i+n_x n_y} T_{i+n_x n_y} = b_i$$

- Linear system: $\mathbf{AT} = \mathbf{b}$, $\mathbf{A}$ sparse with 7 non-zero diagonals



$$\begin{pmatrix} T_1 \\ T_2 \\ T_3 \\ \vdots \\ T_i \\ \vdots \\ T_{n-2} \\ T_{n-1} \\ T_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_i \\ \vdots \\ b_{n-2} \\ b_{n-1} \\ b_n \end{pmatrix}$$

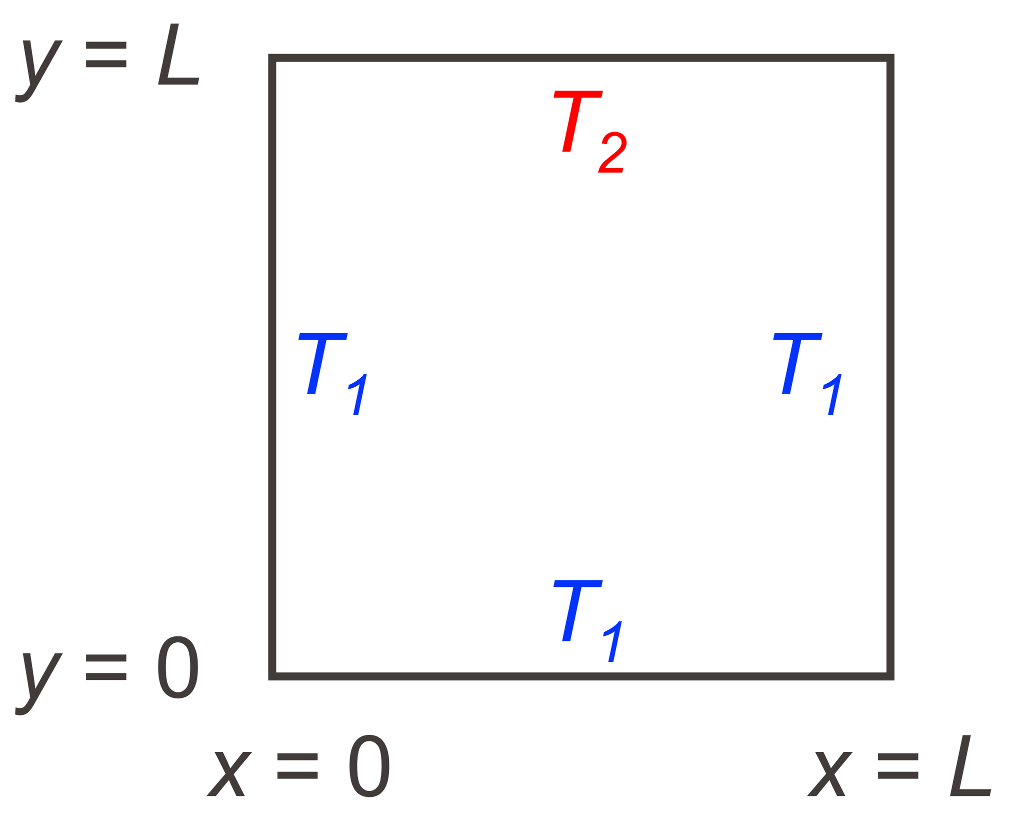
- Advantages of structured meshes:
  - Don't store neighbor connectivity → less memory + faster access to matrix elements
  - Efficient solvers available for this type of matrices

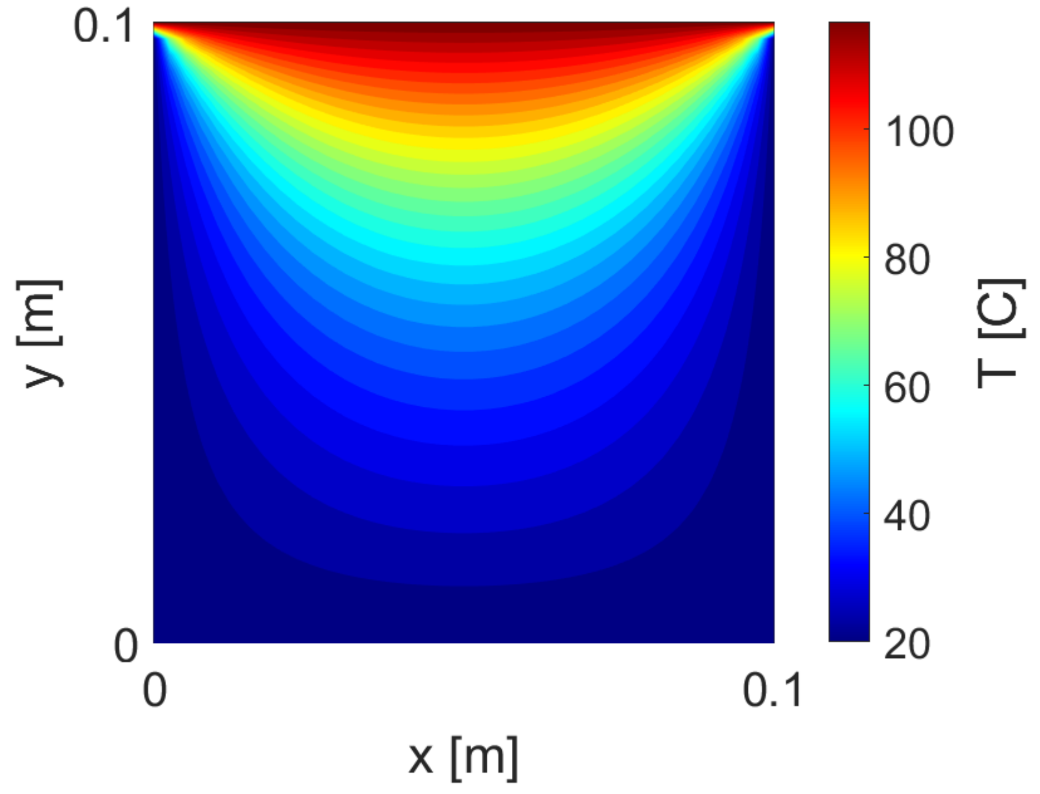
Demo: 2D steady diffusion

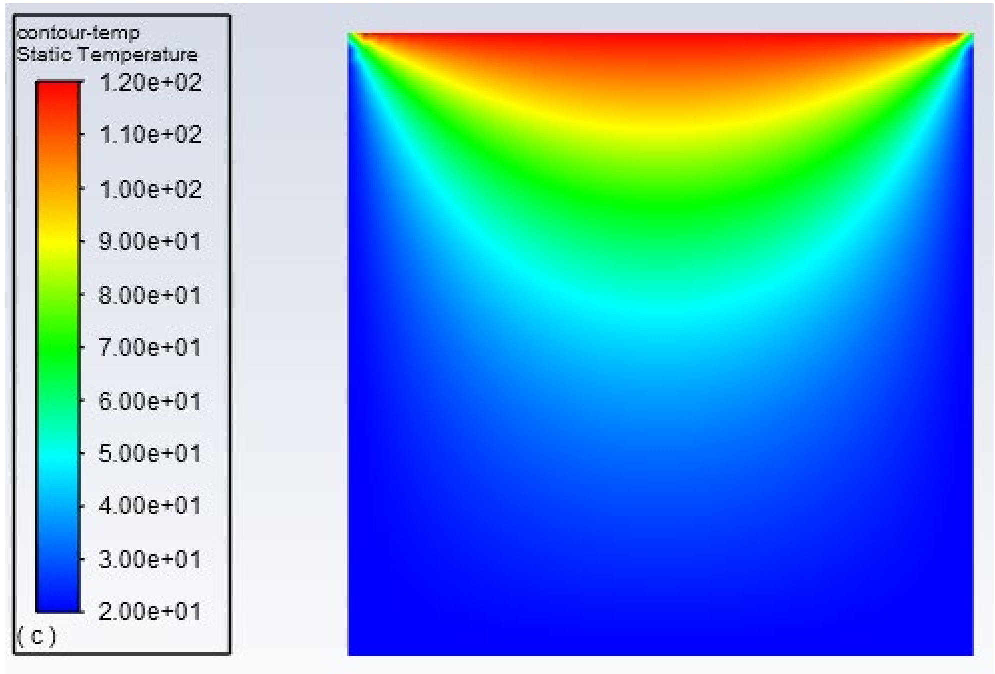
- Cavity filled of a constant-property (independent of *T*) fluid.
  - size: L=0.1 m
  - boundary conditions: T_1=20 °C, T_2=120 °C
  - fluid: air, thermal conductivity k=0.024 W/(m.K)

- Can extend the previous 1D Matlab code to 2D, and solve the heat equation.

- Can also use Fluent to solve the heat equation (and the NS equations, although we already know that **u**=**0** and p=cst).







2D/3D steady diffusion: unstructured mesh

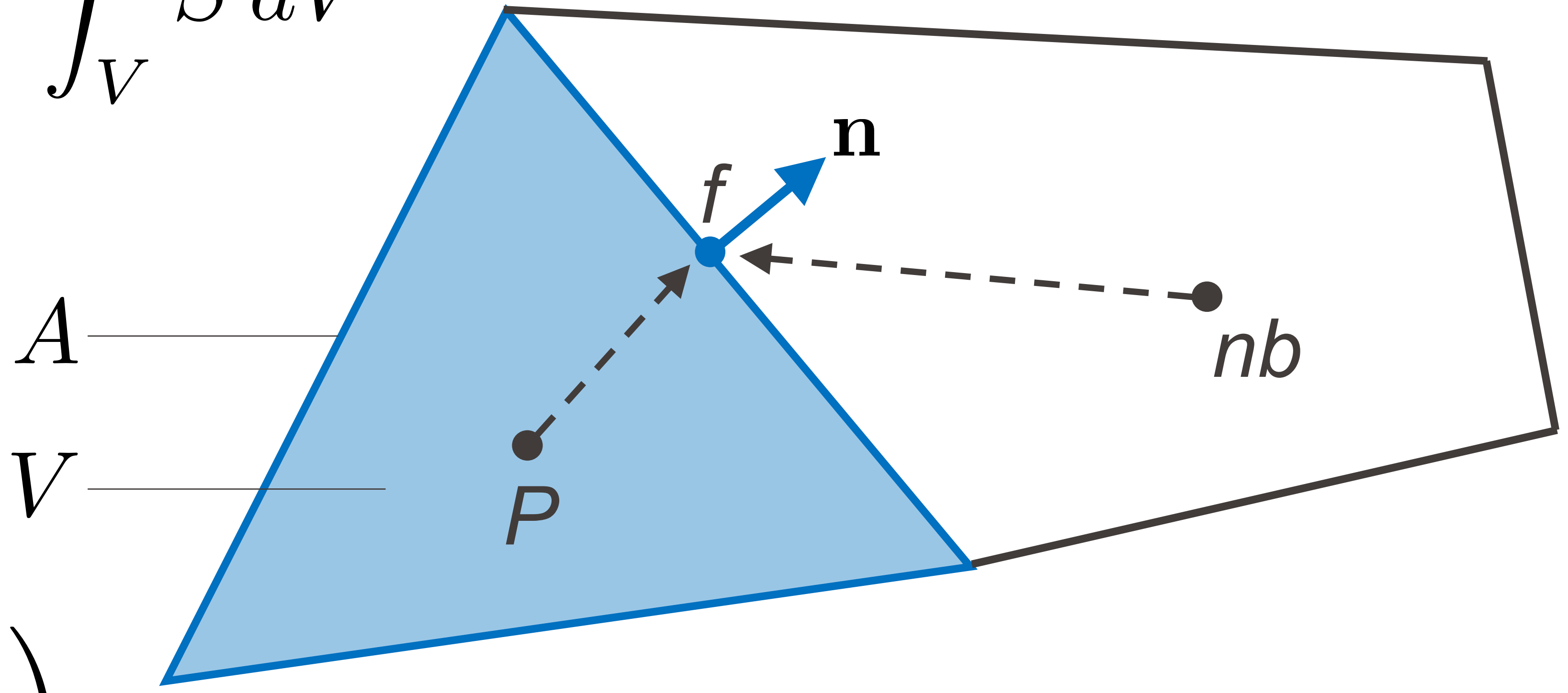
$$div(k\,grad(T)) + S = 0$$

- Integration over CV + divergence theorem

$$0 = \int_V \left(div(k\,grad(T)) + S\right)\,dV = \oint_A k\,grad(T)\cdot\mathbf{n}\,dA + \int_V S\,dV$$

- Need neighbor connectivity + geometric info
- Same type of final algebraic equation…

$$a_P T_P = \sum_{nb} a_{nb} T_{nb} + b$$



$$\begin{bmatrix} 0 & a_{i,j} & & a_{i,j} & 0 & \\ & & 0 & & & \\ a_{i,j} & & & & & a_{i,j} \\ & & & a_{i,j} & & \\ 0 & & 0 & & & \\ & & & & 0 & \\ & 0 & & & & \\ a_{i,j} & & & a_{i,j} & a_{i,j} & \end{bmatrix} \begin{pmatrix} T_1 \\ T_2 \\ T_3 \\ \vdots \\ T_i \\ \vdots \\ T_{n-2} \\ T_{n-1} \\ T_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_i \\ \vdots \\ b_{n-2} \\ b_{n-1} \\ b_n \end{pmatrix}$$

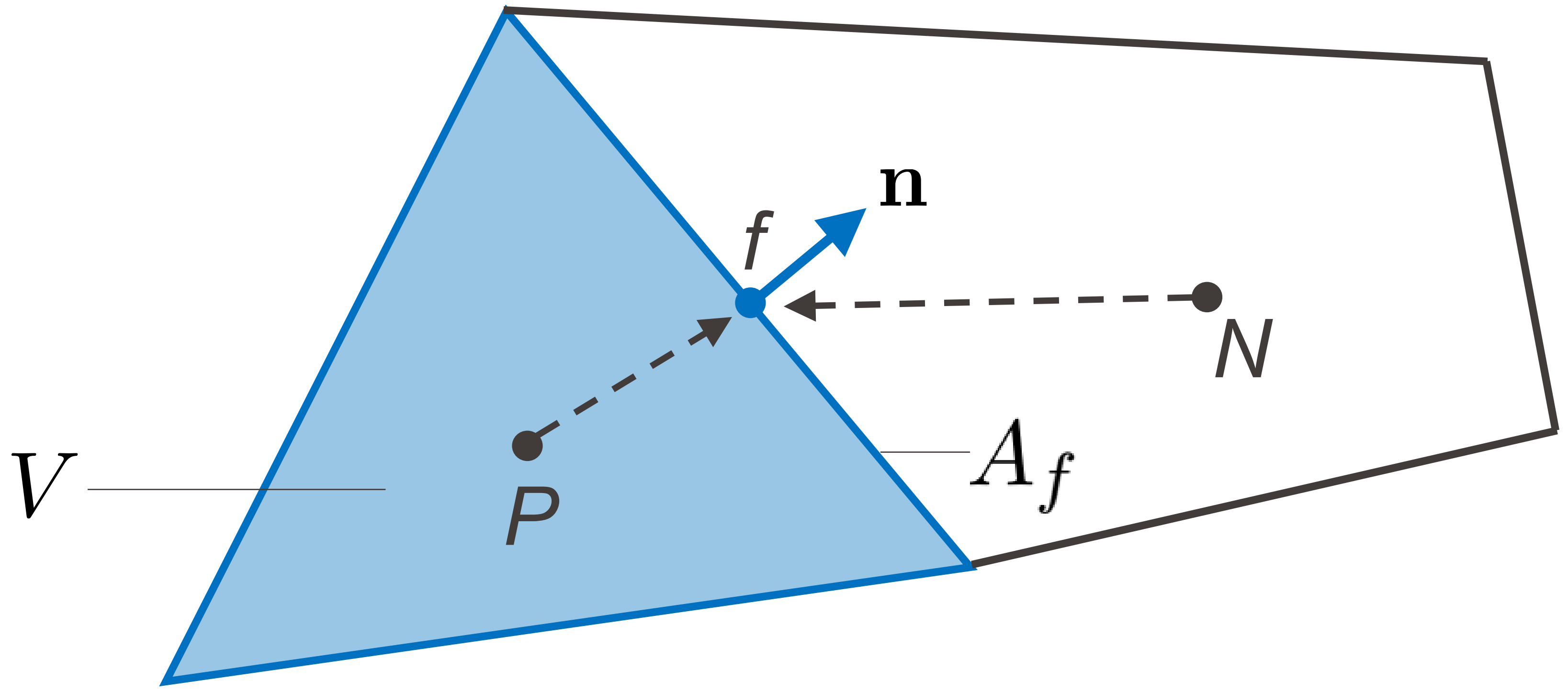
- … but irregular matrix structure

Spatial discretization on unstructured meshes

$$\frac{\partial}{\partial t}\int_{V_i}\rho\phi\,dV+\oint_{A_i}\rho\phi\mathbf{u}\cdot\mathbf{n}\,dA=\oint_{A_i}\Gamma\,grad(\phi)\cdot\mathbf{n}\,dA+\int_{V_i}S\,dV$$

- How does the discretization of the conservation eq. translate from structured to unstructured mesh?
- Volume integrals: same expression

$$\int_{V_i}S\,dV=\overline{S}\,V_i\approx S_P\,V_i$$



- Surface integrals: same idea, more complicated expressions.

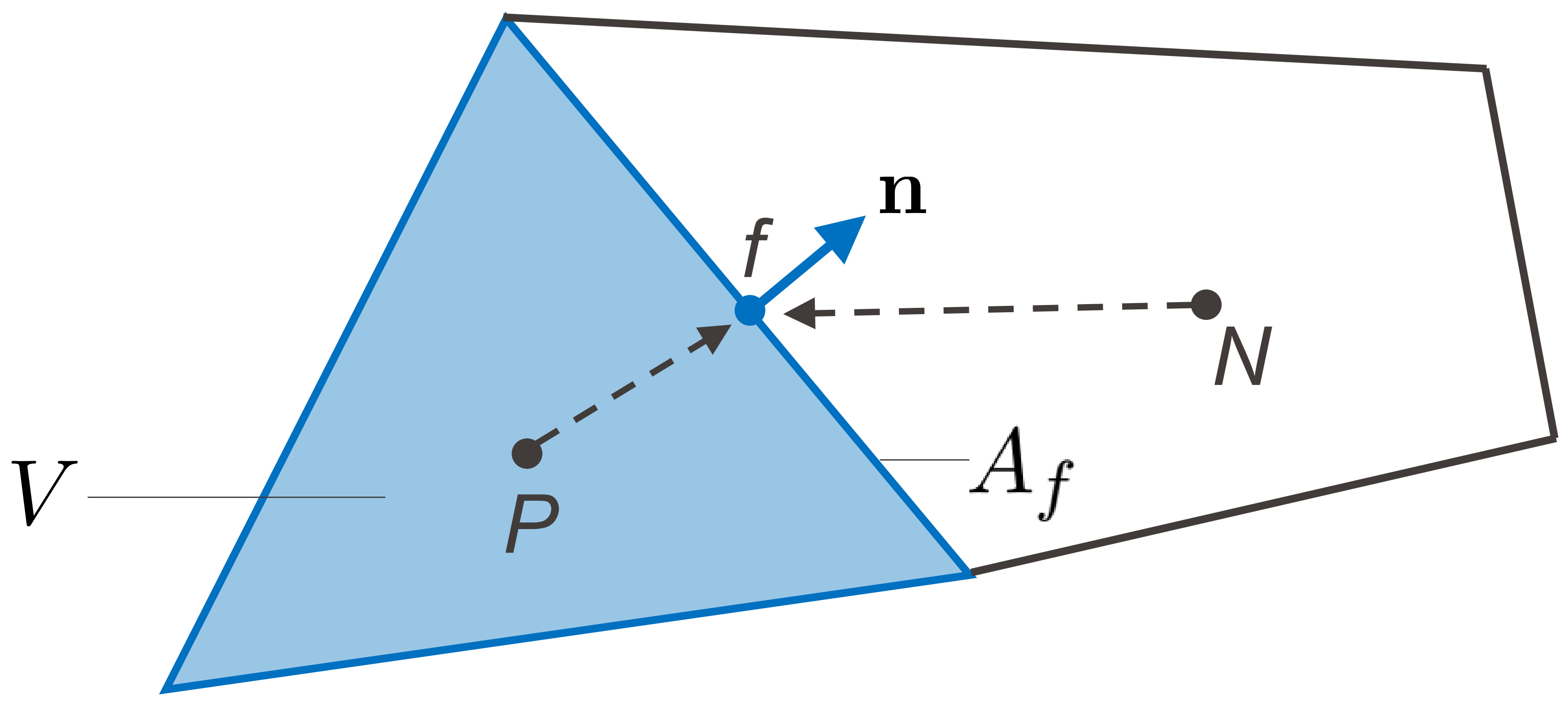
1. Midpoint rule: $$\int_f f\,dA\approx f_f A_f$$ (approximate integral with face value at face center)
2. Interpolate face value f_f from nodal values. For example:

Upwind differencing: $\phi_f\approx\ ?$

Central differencing: $\phi_f\approx\ ?$ $\nabla\phi_f\cdot\mathbf{n}\approx\ ?$

Spatial discretization on unstructured meshes

$$\frac{\partial}{\partial t}\int_{V_i} \rho\phi \, dV + \oint_{A_i} \rho\phi\mathbf{u}\cdot\mathbf{n}\, dA = \oint_{A_i} \Gamma\, grad(\phi)\cdot\mathbf{n}\, dA + \int_{V_i} S\, dV$$



- Surface integrals:

  2. Interpolate face value f_f from nodal values.
  For example, from Taylor expansions:

Upwind differencing: $\phi_f \approx \phi_P$

(depending on sign of normal face velocity $u_f = \mathbf{u}_f \cdot \mathbf{n}$)

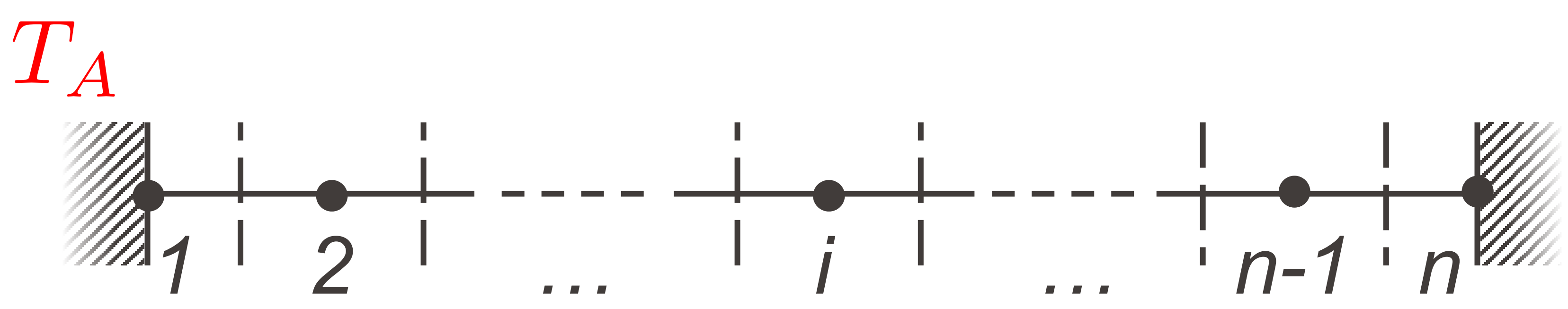
Central differencing: $\phi_f \approx \frac{1}{2}\left(\phi_P + \nabla\phi_P \cdot \overrightarrow{Pf} + \phi_N + \nabla\phi_N \cdot \overrightarrow{Nf}\right)$

(where the gradients require further reconstruction from nodal values)

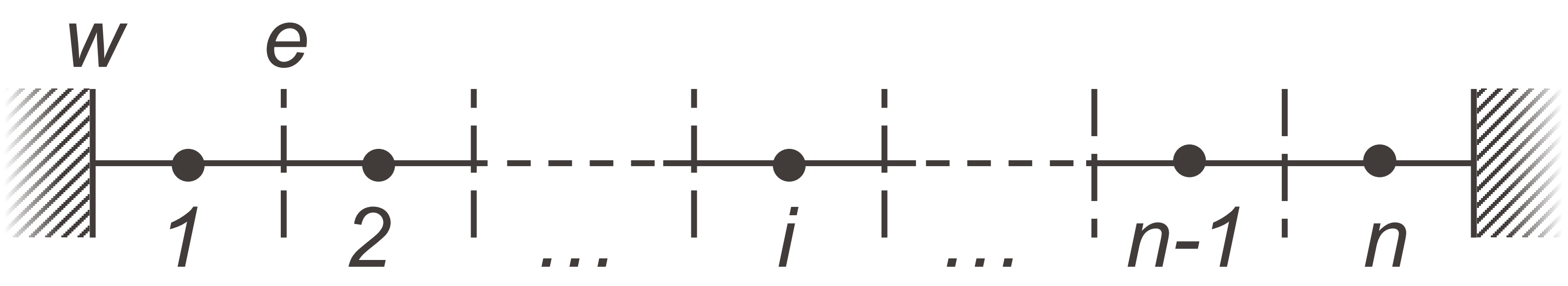
and: $\nabla\phi_f \cdot \mathbf{n} \approx \dfrac{\phi_N - \phi_P}{||\overrightarrow{PN}||}$ (assuming *P*, *f* and *N* are aligned)

Appendix: a note about boundaries

- In the 1D diffusion example, we chose to put the first and last nodes exactly **on the boundary**. This is very convenient when imposing Dirichlet boundary conditions, as one can simply write: $T_1 = T_A$



- In general, it is more common to put all nodes **inside their respective CV**, even for CVs touching a boundary.



In this case, one can show (try it as an exercise) that the Dirichlet boundary condition on the left is implemented as:

$$\left(\frac{k_e}{\delta x_{12}} + \frac{k_1}{\delta x_{w1}}\right) T_1 = \left(\frac{k_e}{\delta x_{12}}\right) T_2 + \left(\frac{k_w}{\delta x_{w1}} T_A + S_1\,\Delta x\right)$$

which is more complicated, but very similar to the general expression in **inner CVs** (CVs not touching the boundaries).

You can try to modify the Matlab code and check that the solution is correct. (Note that the coordinates of the nodes are not the same as in the original example.)