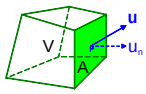
4. The Scalar-Transport Equation

Generic Scalar-Transport Equation



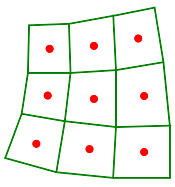
Conservation:

$$\begin{pmatrix}\text{RATE OF CHANGE}\\ \text{inside } V\end{pmatrix} + \begin{pmatrix}\text{FLUX}\\ \text{through boundary of } V\end{pmatrix} = \begin{pmatrix}\text{SOURCE}\\ \text{inside } V\end{pmatrix}$$

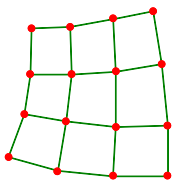
ϕ = concentration (amount per unit mass)

$$\underbrace{\frac{d}{dt}(mass\times\phi)}_{\text{rate of change}} + \sum_{faces}\Big(\underbrace{mass\ flux\times\phi}_{\text{advection}} \underbrace{-\Gamma\frac{\partial\phi}{\partial n}A}_{\text{diffusion}}\Big) = \underbrace{S}_{\text{source}}$$

Location of Storage Nodes

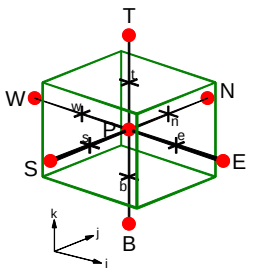


Cell-centred



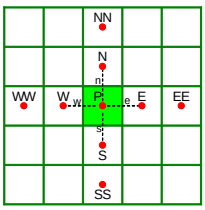
Cell-vertex

Cell Indexing (Structured Meshes)

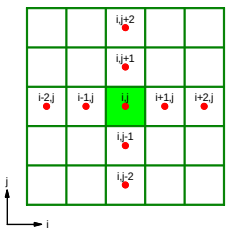


Cell Indexing (Structured Meshes)

Local



Global

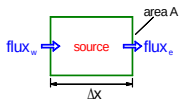


1-d Advection-Diffusion Equation

- Simple analysis
- Hand solution
- Straightforward generalisation to 2-d/3-d
- Coordinatewise discretisation in practice
- Many important theoretical problems are 1-d

Steady 1-d Advection-Diffusion Equation

Conservation: $flux_e - flux_w = source$



Fluxes:

Advection: $(\rho u A)\phi$ (mass flux × concentration)

Diffusion: $-\Gamma A \frac{d\phi}{dx}$ (diffusivity × area × gradient)

Differential form:

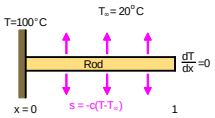
$$\frac{d}{dx}(flux) = source\ per\ unit\ length$$

$$\frac{d}{dx}(\rho u A\phi - \Gamma A\frac{d\phi}{dx}) = s$$

Discretising Diffusion and Source

Example

A thin rod has length 1 m and cross-section 1 cm × 1 cm. The left-hand end is kept at 100 C, whilst the right-hand end is insulated.



The heat flux across any section of area A is given by

$$-kA\frac{dT}{dx}$$

where the conductivity k = 1000 W m^{-1} K^{-1}.

The rod is allowed to cool along its length at a rate proportional to its difference from the ambient temperature (Newton's law of cooling); i.e. the heat source per unit length is:

$$s = -c(T - T_\infty)$$

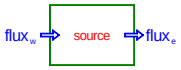
where the ambient temperature T_∞ = 20 ºC and the coefficient c = 2.5 W m^{-1} K^{-1}.

(a) Write down and solve the differential equation satisfied by the temperature.

(b) Divide the rod into 5 control sections with nodes at the centre of each section and carry out a finite-volume analysis to find the temperature along the rod.

Differential Equation and Exact Solution

$flux_e - flux_w = source$



$$\left(-kA\frac{dT}{dx}\right)_e - \left(-kA\frac{dT}{dx}\right)_w = -c(T-T_\infty)\Delta x$$

$$\frac{\left(-kA\frac{dT}{dx}\right)_e - \left(-kA\frac{dT}{dx}\right)_w}{\Delta x} = -c(T-T_\infty)$$

$$\frac{d}{dx}\left(-kA\frac{dT}{dx}\right) = -c(T-T_\infty)$$

$T(0)=100, \quad T'(1)=0$

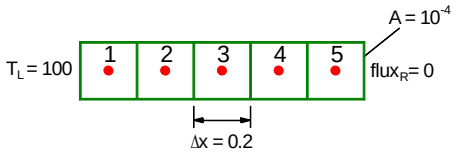
$$\frac{d^2T}{dx^2} - \frac{c}{kA}T = -\frac{c}{kA}T_\infty$$

$$\frac{d^2T}{dx^2} - 25T = -500$$

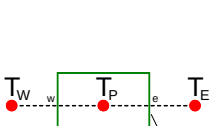
$$T = \underbrace{Ae^{5x} + Be^{-5x}}_{\text{complementary function}} + \underbrace{20}_{\text{particular integral}}$$

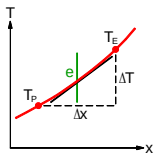
$A=0.0036, \quad B=79.9964$

Finite-Volume Decomposition



Discretisation of Fluxes





$$flux_e = -kA\left.\frac{dT}{dx}\right|_e \rightarrow -kA\left(\frac{T_E - T_P}{\Delta x}\right)$$

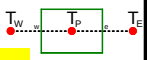
Finite-Volume Discretisation

Conservation: $flux_e - flux_w = source$

Fluxes:

interior faces: $flux_e = -kA\left.\frac{dT}{dx}\right|_e \rightarrow -kA\left(\frac{T_E - T_P}{\Delta x}\right)$

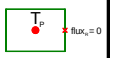
$\rightarrow -D(T_E - T_P)$ $\quad D = \frac{kA}{\Delta x} = 0.5$



left boundary: $flux_L = -kA\left.\frac{dT}{dx}\right|_L \rightarrow -kA\left(\frac{T_P - T_L}{\frac{1}{2}\Delta x}\right)$

$\rightarrow -2D(T_P - T_L)$

right boundary: $flux_R = 0$

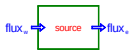


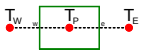
Source: $source = -c(T_P - T_\infty)\Delta x \rightarrow b_P + s_P T_P$

$b_P = cT_\infty \Delta x = 10, \quad s_P = -c\Delta x = -0.5$

Finite-Volume Discretisation

$flux_e - flux_w = source$





Interior cells: $-0.5(T_E - T_P) \quad +0.5(T_P - T_W) = 10 - 0.5T_P$

$-0.5T_W + 1.5T_P - 0.5T_E = 10$

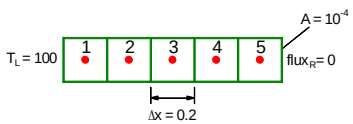
Left boundary: $-0.5(T_E - T_P) \quad +1.0(T_P - 100) = 10 - 0.5T_P$

$2.0T_P - 0.5T_E = 110$

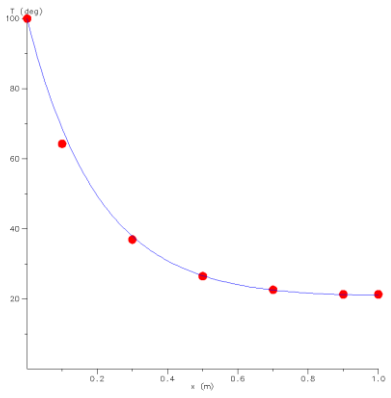
Right boundary: $0 \quad +0.5(T_P - T_W) = 10 - 0.5T_P$

$-0.5T_W + 1.0T_P = 10$

Assembled Equations



$$\begin{pmatrix} 2 & -0.5 & 0 & 0 & 0 \\ -0.5 & 1.5 & -0.5 & 0 & 0 \\ 0 & -0.5 & 1.5 & -0.5 & 0 \\ 0 & 0 & -0.5 & 1.5 & -0.5 \\ 0 & 0 & 0 & -0.5 & 1 \end{pmatrix} \begin{pmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \end{pmatrix} = \begin{pmatrix} 110 \\ 10 \\ 10 \\ 10 \\ 10 \end{pmatrix}$$



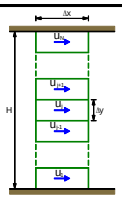
Points to Note

- Each control volume gives one equation
- Each equation is a linear equation connecting the central node and nodes on either side
- Boundary values are incorporated into the source term
- The resulting matrix is tri-diagonal

$$\left(\quad\right)\Phi = \mathbf{b}$$

Example

A 2-d finite-volume calculation is to be undertaken for fully-developed, laminar flow between stationary, plane, parallel walls. A streamwise pressure gradient $\mathrm{d}p/\mathrm{d}x = -G$ is imposed and the fluid viscosity is μ. The depth of the channel, H, is divided into N equally-sized cells of dimension $\Delta x \times \Delta y \times 1$ as shown, with the velocity u stored at cell centres.



(a) What are the boundary conditions on velocity?

(b) What is the net pressure force on a single cell?

(c) Using a finite-difference approximation for velocity gradient, find expressions for the viscous forces on upper and lower faces of the j^{th} cell in terms of the nodal velocities $\{u_j\}$. (You will need to deal separately with interior cells and the boundary cells $j = 1$ and $j = N$.)

(d) By balancing pressure and viscous forces set up the finite-volume equations for velocity.

(e) Solve for the nodal velocities in the case $N = 6$, leaving your answers as multiples of $U_0 = GH^2/\mu$. (You are advised to note the symmetry of the problem.)

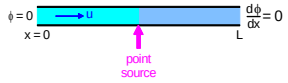
(f) Using your numerical solution, find the volume flow rate (per unit span) Q in terms of U_0 and H.

(g) Find the wall shear stress, τ_w.

(h) Compare your answers to (e), (f), (g) with the exact solution for plane Poiseuille flow.

Discretising Advection (Part 1)

Example



A pipe of cross-section $A = 0.01$ m^2 and length $L = 1$ m carries water (density $\rho = 1000$ kg m^{-3}) at velocity $u = 0.1$ m s^{-1}.

A faulty valve introduces a reactive chemical into the pipe half-way along its length at a rate of 0.01 kg s^{-1}. The diffusivity of the chemical in water is $\Gamma = 0.1$ kg m^{-1} s^{-1}. The chemical is subsequently broken down at a rate proportional to its concentration ϕ (mass of chemical per unit mass of water), this rate amounting to $-\gamma\phi$ per metre, where $\gamma = 0.5$ kg m^{-1} s^{-1}.

Assuming that the downstream boundary condition is $d\phi/dx = 0$, set up a finite-volume calculation with 7 cells to estimate the concentration along the pipe using:
(a) **Central**
(b) **Upwind**
differencing schemes for advection.

Fluxes and Sources

Concentration ϕ (= mass of chemical per mass of water)

Flux (= rate of transport across a surface)

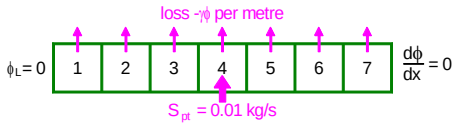
advection: $C\phi$ $\quad C = \rho u A$

diffusion: $-\Gamma A \dfrac{d\phi}{dx}$

Source

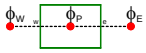
- point source S_{pt} at $x = 0.5$ m
- loss by chemical breakdown $-\gamma\phi$ per unit length

Finite-Volume Decomposition



Finite-Volume Discretisation

$flux_e - flux_w = source$



$flux = C\phi - \Gamma A\frac{d\phi}{dx}$

$flux_e = C\phi_e - D(\phi_E - \phi_P)$

$flux_w = C\phi_w - D(\phi_P - \phi_W)$

$C = \rho uA = 1.0$

$D = \frac{\Gamma A}{\Delta x} = 0.007$

$source = \underbrace{S_{pt}}_{\text{cell 4 only}} - \gamma\phi_P\Delta x$

$source = b_P + s_P\phi_P$

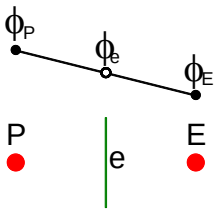
$b_P = 0.01$ (cell 4)

$s_P = -0.071$

$$\underbrace{C\phi_e - C\phi_w}_{advection} \quad \underbrace{-D\phi_W + 2D\phi_P - D\phi_E}_{diffusion} \quad = \quad \underbrace{b_P + s_P\phi_P}_{source}$$

An approximation for **cell-face** values ϕ_e and ϕ_w is called an **advection scheme** or **advection-differencing scheme**

Central Differencing

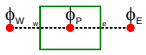


$\phi_e = \frac{1}{2}(\phi_P + \phi_E)$

Finite-Volume Discretisation:

Central Differencing For Advection

$$\underbrace{C\phi_e - C\phi_w}_{advection} \quad \underbrace{-D\phi_W + 2D\phi_P - D\phi_E}_{diffusion} \quad = \quad \underbrace{b_P + s_P\phi_P}_{source}$$



Central differencing: $C\phi_e - C\phi_w = C\frac{1}{2}(\phi_P + \phi_E) - C\frac{1}{2}(\phi_W + \phi_P)$

$$= \frac{C}{2}(\phi_E - \phi_W)$$

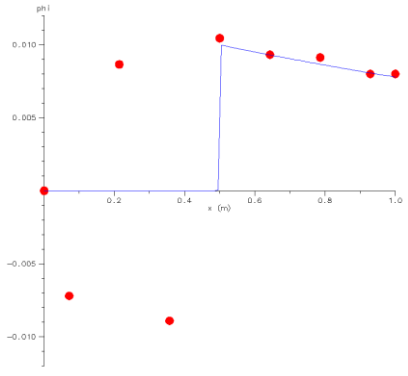
$$-(\frac{C}{2} + D)\phi_W + (2D - s_P)\phi_P - (-\frac{C}{2} + D)\phi_E = b_P$$

$$-0.507\phi_W + 0.085\phi_P + 0.493\phi_E = \underbrace{0.01}_{cell\ 4\ only}$$

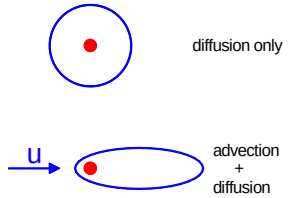
Assembled Equations

Central Advection Scheme

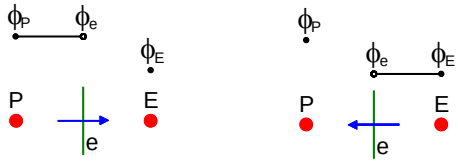
$$\begin{pmatrix} 0.592 & 0.493 & 0 & 0 & 0 & 0 & 0 \\ -0.507 & 0.085 & 0.493 & 0 & 0 & 0 & 0 \\ 0 & -0.507 & 0.085 & 0.493 & 0 & 0 & 0 \\ 0 & 0 & -0.507 & 0.085 & 0.493 & 0 & 0 \\ 0 & 0 & 0 & -0.507 & 0.085 & 0.493 & 0 \\ 0 & 0 & 0 & 0 & -0.507 & 0.085 & 0.493 \\ 0 & 0 & 0 & 0 & 0 & -0.507 & 0.578 \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \\ \phi_5 \\ \phi_6 \\ \phi_7 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0.01 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$



Directional Bias



Upwind Differencing

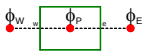


$\phi_e = \phi_P$ (if $u > 0$)

$\phi_e = \phi_E$ (if $u < 0$)

Upwind Differencing

$$\underbrace{C\phi_e - C\phi_w}_{advection} \quad \underbrace{-D\phi_W + 2D\phi_P - D\phi_E}_{diffusion} = \underbrace{b_P + S_P\phi_P}_{source}$$



Upwind differencing: $C\phi_e - C\phi_w = C(\phi_P - \phi_W)$

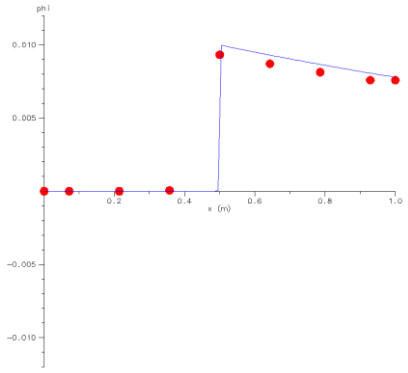
$$-(C+D)\phi_W + (C+2D-S_P)\phi_P - D\phi_E = b_P$$

$$-1.007\phi_W + 1.085\phi_P - 0.007\phi_E = \underbrace{0.01}_{cell\ 4\ only}$$

Assembled Equations

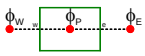
- Upwind Advection Scheme

$$\begin{pmatrix} 1.092 & -0.007 & 0 & 0 & 0 & 0 & 0 \\ -1.007 & 1.085 & -0.007 & 0 & 0 & 0 & 0 \\ 0 & -1.007 & 1.085 & -0.007 & 0 & 0 & 0 \\ 0 & 0 & -1.007 & 1.085 & -0.007 & 0 & 0 \\ 0 & 0 & 0 & -1.007 & 1.085 & -0.007 & 0 \\ 0 & 0 & 0 & 0 & -1.007 & 1.085 & -0.007 \\ 0 & 0 & 0 & 0 & 0 & -1.007 & 1.078 \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \\ \phi_5 \\ \phi_6 \\ \phi_7 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0.01 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$



Simple Advection Schemes Compared

$$-a_W\phi_W + a_P\phi_P - a_E\phi_E = b_P$$



Central differencing

$$a_W = \tfrac{1}{2}C + D$$
$$a_E = -\tfrac{1}{2}C + D$$
$$a_P = 2D - s_P$$

OK $\Leftrightarrow$ $\mathrm{Pe} \le 2$

$$\mathrm{Pe} = \frac{C}{D} = \frac{\rho u \Delta x}{\Gamma}$$ **Peclet number**

Theoretically, high accuracy ...
... but unphysical “wiggles”

Upwind differencing

$$a_W = C + D$$
$$a_E = D$$
$$a_P = C + 2D - s_P$$

(if $C > 0$)

Lower accuracy ...
... but no “wiggles”

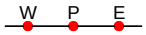
Finite-Volume Analysis: 1-d

Conservation:

$$flux_e - flux_w = source$$

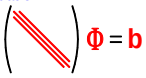
$$flux = C\phi - \Gamma A \frac{d\phi}{dx}$$

Discretisation:



$$-a_W\phi_W + a_P\phi_P - a_E\phi_E = b_P$$

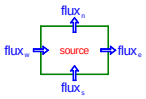
Assembled matrix equation:



$$\mathbf{A\Phi = b}$$

Finite-Volume Analysis: 2-d

Conservation:

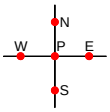


$$flux_e - flux_w + flux_n - flux_s = source$$

$$flux_{e,w} = (\rho u A\phi - \Gamma A \frac{\partial\phi}{\partial x})_{e,w}$$

$$flux_{n,s} = (\rho v A\phi - \Gamma A \frac{\partial\phi}{\partial y})_{n,s}$$

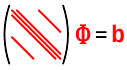
Discretisation:



$$-a_S\phi_S - a_W\phi_W + a_P\phi_P - a_E\phi_E - a_N\phi_N = b_P$$

$$a_P\phi_P - \sum a_F\phi_F = b_P$$

Assembled matrix equation:



$$\mathbf{A\Phi = b}$$

General Discretisation Properties

Discretisation Properties

- Consistency
- Conservativeness
- Transportiveness
- Boundedness
- Stability
- Accuracy (“order”)

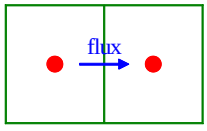
Consistent

Definition: an algebraic approximation is **consistent** if it tends to the exact governing equation as the mesh size tends to zero

e.g. $\frac{\phi_E - \phi_P}{\Delta x} \leftarrow \frac{\mathrm{d}\phi}{\mathrm{d}x}$

Conservativeness

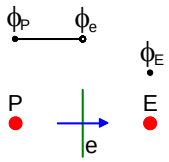
Definition: flux **out** of one cell = flux **into** adjacent cell



- **Fluxes** are worked out by **face**, not by cell
- Automatically built into the finite-volume method

Transportiveness

Definition: upstream-biased



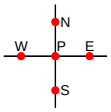
Boundedness

Definition: for advection-diffusion **without sources**:

(1) ϕ_P must lie between minimum and maximum values at surrounding nodes

(2) ϕ = constant must be a possible solution

$$a_P\phi_P - \sum a_F\phi_F = 0$$



Requirements:

(1) Positive coefficients: $a_P, a_F \geq 0$

(2) Sum of neighbouring coefficients: $a_P = \sum a_F$

Stability

Definition:

– small errors do not grow in the course of the calculation

Requirement:

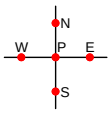
– negative-slope linearisation of the source term ("negative feedback")

$$source = b_P + s_P\phi_P, \qquad s_P \leq 0$$

Summary of Conditions on Matrix Coefficients

$$source = b_P + s_P \phi_P$$

$$a_P \phi_P - \sum a_F \phi_F = b_P$$



Positive coefficients:

$a_F \geq 0$ for all F

Negative-slope linearisation of the source term:

$s_P \leq 0$

Sum of neighbouring coefficients:

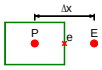
$a_P = \sum a_F - s_P$

Accuracy ("Order")

- General definition:

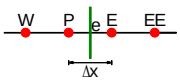
  order n ⇔ error ∝ $(\Delta x)^n$ (as $\Delta x \to 0$)

- Determined by Taylor-series expansion (about a cell face)



- Common advection schemes:
  - upwind: 1st order
  - central: 2nd order
  - QUICK: 3rd order

Example



Consider the uniform, one-dimensional arrangement of nodes shown. Face e lies half way between P and E nodes.

(a) Show that the central-differencing schemes

$$\phi_e \approx \tfrac{1}{2}(\phi_P + \phi_E) \qquad \left.\frac{d\phi}{dx}\right|_e \approx \frac{\phi_E - \phi_P}{\Delta x}$$

are second-order accurate approximations for ϕ_e and $(d\phi/dx)_e$ respectively.

(b) Making use of the *W* and *EE* nodes also, find symmetric fourth-order-accurate approximations for ϕ_e and $(d\phi/dx)_e$.

Discretising Advection (Part 2)

Exponential Scheme

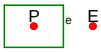
Exact solution of the 1-d advection-diffusion equation

(with constant coefficients and without sources)

$$flux = C\phi - \Gamma A\frac{d\phi}{dx} = constant$$

$$flux_e = C\left(\frac{e^{Pe}\phi_P - \phi_E}{e^{Pe}-1}\right)$$

$$flux_e - flux_w = a_P\phi_P - \Sigma a_F\phi_F$$



$$C = \rho uA, \quad D = \frac{\Gamma A}{\Delta x}, \quad \mathrm{Pe} = \frac{C}{D}$$

$$a_W = \frac{Ce^{Pe}}{e^{Pe}-1}, \quad a_E = \frac{C}{e^{Pe}-1}$$
$$a_P = a_E + a_W$$

Hybrid Scheme

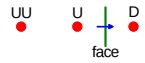
More computationally-efficient approximation to the exponential scheme.

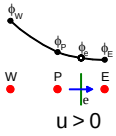
Amounts to:

- Central differencing if |Pe| ≤ 2
- Upwind differencing (with no diffusion) if |Pe| > 2

QUICK

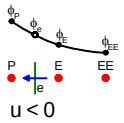
- **3-point**, upwind-biased scheme





- Fits a quadratic through 3 nodes

$$\phi_{face} = -\tfrac{1}{8}\phi_{UU} + \tfrac{3}{4}\phi_U + \tfrac{3}{8}\phi_D$$



- 3rd-order accurate ... but not bounded

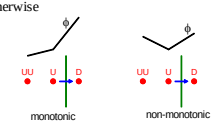
Flux-Limited Schemes

- **3-point**, **upwind-biased** schemes, with **solution-dependent** limiters

$$\phi_{face} = \text{function}(\phi_{UU}, \phi_U, \phi_D)$$

- General form: $\phi_{face} = \begin{cases} \phi_U + \tfrac{1}{2}\psi(r)(\phi_D - \phi_U) & \text{if monotonic} \\ \phi_U & \text{otherwise} \end{cases}$

$$r = \frac{\phi_U - \phi_{UU}}{\phi_D - \phi_U}$$



- Non-linear ($\Rightarrow$ iteration necessary)
- **T**otal-**V**ariation-**D**iminishing

Scheme	**$\psi(r)$ (for $r > 0$)**	
UMIST	$\min\{2,\ 2r,\ \tfrac{1}{4}(1+3r),\ \tfrac{1}{4}(3+r)\}$	Upstream Monotonic Interpolation for Scalar Transport (Lien and Leschziner, 1993)
Harmonic	$\frac{2r}{1+r}$	Van Leer (1974)
Min-mod	$\min\{r,1\}$	Roe (1985)
Van Albada et al.	$\frac{r+r^2}{1+r^2}$	Van Albada et al., (1982)

In all cases $\psi(r)$ is taken as 0 if non-monotonic ($r < 0$)

Example

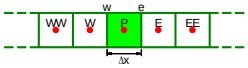
The Van Leer harmonic advection scheme is given by

$$\phi_{face} = \phi_U + \tfrac{1}{2}\psi(r)(\phi_D - \phi_U)$$

$$\psi(r) = \begin{cases} \dfrac{2r}{1+r} & \text{if } r > 0 \\ 0 & \text{otherwise} \end{cases} \qquad r = \frac{\phi_U - \phi_{UU}}{\phi_D - \phi_U}$$

where nodes UU, U and D are determined by flow direction as shown.

A local arrangement of nodes and faces in a 1-d finite-volume mesh with standard geographical notation and uniform mesh spacing Δx is shown below.



The values of a scalar ϕ at these nodes are:

$\phi_{WW} = 2, \quad \phi_W = 4, \quad \phi_P = 6, \quad \phi_E = 7, \quad \phi_{EE} = 6.$

Using the Van Leer harmonic advection scheme, calculate the value of ϕ on the west (w) and east (e) cell faces if the velocity throughout the domain is:
(a) positive;
(b) negative.

Implementation of Higher-Order Schemes

$$\sum_{faces}\left(\begin{array}{c} mass\ flux \times \phi \\ advection \end{array} - \Gamma \frac{\partial \phi}{\partial n} A \right) = \begin{array}{c} S \\ source \end{array}$$
$$\begin{array}{c} \\ diffusion \end{array}$$

$$\sum_{faces}[C_f\phi_f + D_f(\phi_P - \phi_F)] = b_P + s_P\phi_P$$

$$\sum_{faces} C_f = 0 \qquad \sum_{faces}[C_f(\phi_f - \phi_P) + D_f(\phi_P - \phi_F)] = b_P + s_P\phi_P$$

Write the cell-face value as **upwind** + **correction**: $\phi_f = \underbrace{\phi_U}_{upwind} + \underbrace{(\phi_f - \phi_U)}_{deferred\ correction}$

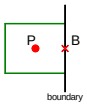
The **upwind** part always gives positive coefficients and diagonal dominance

$$\begin{aligned} C_f(\phi_f - \phi_P) &= \underbrace{C_f(\phi_U - \phi_P)}_{upwind} + \underbrace{C_f(\phi_f - \phi_U)}_{correction} \\ &= \max(-C_f, 0)(\phi_P - \phi_F) + C_f(\phi_f - \phi_U) \end{aligned}$$

The **deferred correction** is transferred to the source term and treated explicitly (i.e. not updated until the next iteration):

$$\sum_F a_F(\phi_P - \phi_F) = b_P + s_P\phi_P - \underbrace{\sum_{faces} C_f(\phi_f - \phi_U)}_{deferred\ correction} \qquad a_F = \max(-C_f, 0) + D_f$$

Implementation of Boundary Conditions



Boundary fluxes are transferred to the **source** term:

$$\sum_{\substack{not \\ boundary}} flux + flux_{boundary} = source$$

$$\downarrow$$

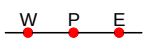
$$a_P\phi_P - \sum_{\substack{not \\ boundary}} a_F\phi_F = b_P - flux_{boundary}$$

Matrix Solution Algorithms

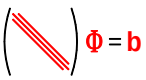
Form of the Matrix Equations

$$a_P\phi_P - \sum_F a_F\phi_F = b_P$$

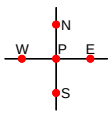
1-d



$$-a_W\phi_W + a_P\phi_P - a_E\phi_E = b_P$$



2-d



$$-a_S\phi_S - a_W\phi_W + a_P\phi_P - a_E\phi_E - a_N\phi_N = b_P$$

Characteristics of the Equations

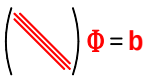
- **Fluid-flow equations:**
  - non-linear
  - Coupled
- **Matrix equations:**
  - sparse
  - Banded
- **Solution methods:**
  - iterative

Common Solution Methods

- Gaussian elimination
- Gauss-Seidel
- Line Gauss-Seidel

Gaussian Elimination

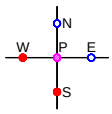
- **Direct**, not iterative
- Sequence of **row operations** aiming to produce an **upper-triangular matrix**
- Tends to "**fill in**" sparse matrices
- Important exception: tri-diagonal systems
  (basis of the **tri-diagonal matrix algorithm**)

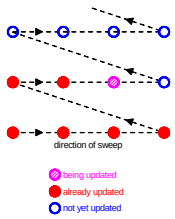


Gauss-Seidel

$$-a_S\phi_S - a_W\phi_W + a_P\phi_P - a_E\phi_E - a_N\phi_N = b_P$$

$$\phi_P = \frac{1}{a_P}(b_P + a_S\phi_S^* + a_W\phi_W^* + a_E\phi_E^* + a_N\phi_N^*)$$





Gauss-Seidel Example (i)

$$\begin{pmatrix} 4 & -1 & 0 & 0 \\ -1 & 4 & -1 & 0 \\ 0 & -1 & 4 & -1 \\ 0 & 0 & -1 & 4 \end{pmatrix} \begin{pmatrix} A \\ B \\ C \\ D \end{pmatrix} = \begin{pmatrix} 2 \\ 4 \\ 6 \\ 13 \end{pmatrix}$$

Gauss-Seidel Example (ii)

$$\begin{pmatrix} 1 & -4 & 0 & 0 \\ -4 & 1 & -4 & 0 \\ 0 & -4 & 1 & -4 \\ 0 & 0 & -4 & 1 \end{pmatrix} \begin{pmatrix} A \\ B \\ C \\ D \end{pmatrix} = \begin{pmatrix} -7 \\ -14 \\ -21 \\ -8 \end{pmatrix}$$

Gauss-Seidel Overview

- **Iterative**
- **Simple** to code
- Minimal **storage** requirements
- **Convergence slow** for large matrices
- Constraints on matrix elements (e.g. **diagonal dominance**)

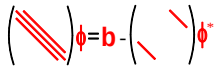
$$\sum_{j \neq i} |a_{ij}| \leq a_{ii}$$

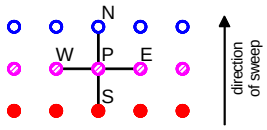
Line Gauss-Seidel

(Line-Iterative Procedures, LIP)

$$-a_S\phi_S - a_W\phi_W + a_P\phi_P - a_E\phi_E - a_N\phi_N = b_P$$

$$-a_W\phi_W + a_P\phi_P - a_E\phi_E = b_P + a_S\phi_S^* + a_N\phi_N^*$$





Line Gauss-Seidel - Overview

- **Iterative**
- Repeated use of **tri-diagonal solver**:
  - **whole line** updated in one go
  - rapid propagation across domain
- Applicable to **structured meshes** only
- Basis of many research codes

Convergence Criteria

$$a_P\phi_P - \sum_F a_F\phi_F = b_P$$

Single-cell error:

residual: $res = a_P\phi_P - \sum_F a_F\phi_F - b_P$

Global error:

sum of absolute residuals: $\sum_{cells}|res|$

root-mean-square error: $\sqrt{\frac{1}{N}\sum_{cells}(res)^2}$

Convergence criteria:

error < tolerance
or
error < fraction of initial error

Under-Relaxation

- The governing equations are coupled and non-linear
- Matrix coefficients change at every iteration
- The iterative solution method may be numerically unstable
- **Under-relaxation** tries to stabilise an iterative procedure by **reducing the change** in variables at each iteration

Under-Relaxation in CFD

Standard linearised equation: $a_P\phi_P - \sum a_F\phi_F = b_P$

Rearrange as the **change** in ϕ: $\phi_P = \dfrac{\sum a_F\phi_F + b_P}{a_P}$

$$\phi_P = \phi_P^{prev} + \left(\frac{\sum a_F\phi_F + b_P}{a_P} - \phi_P^{prev}\right)$$

Under-relax the change: $\phi_P = \phi_P^{prev} + \alpha\left(\dfrac{\sum a_F\phi_F + b_P}{a_P} - \phi_P^{prev}\right)$

Rearrange back: $a_P\phi_P - \sum \alpha a_F\phi_F = \alpha b_P + (1-\alpha)a_P\phi_P^{prev}$

Incorporate by **modifying coefficients**:

$$a_F \rightarrow \alpha a_F$$
$$b_P \rightarrow \alpha b_P + (1-\alpha)a_P\phi_P^{prev}$$

Summary (1)

- The generic scalar-transport equation for a particular control volume has the form
  rate of change + net outward flux = source
- **Flux** ≡ rate of transport through a surface and consists of:
  advection: transport with the flow (other authors prefer *convection*)
  diffusion: net transport by random molecular or turbulent fluctuations
- Discretisation of the (steady) scalar-transport equation yields an equation of form
  $$a_P\phi_P - \sum_F a_F\phi_F = b_P$$
  for each control volume, where the summation is over adjacent nodes
- The collection of these simultaneous equations on a structured mesh yields a **matrix equation** with limited bandwidth (i.e. few non-zero diagonals), typically solved by iterative methods such as Gauss-Seidel or line-Gauss-Seidel

Summary (2)

- **Source** terms are linearised as

$$b_P + s_P \phi_P , \quad s_P \le 0$$

- **Diffusive** fluxes are usually discretised by central differencing; e.g.

$$-\Gamma \left.\frac{\partial \phi}{\partial x}\right|_e A \quad \rightarrow \quad -\frac{\Gamma A}{\Delta x}(\phi_E - \phi_P)$$

- **Advection** schemes are means of approximating ϕ on cell faces in order to compute advective fluxes. They include Upwind, Central, Exponential, Hybrid, QUICK, and various flux-limited schemes
- General desirable properties for a numerical scheme:
  - **consistency**
  - **conservativeness**
  - **transportiveness**
  - **boundedness**
  - **stability**
  - **accuracy / high order**

Summary (3)

- **Boundedness** and **stability** impose certain constraints on the discretisation:

$a_F \ge 0$ ("positive coefficients")

$s_P \le 0$ ("negative feedback in the source term")

$a_P = \sum_F a_F - s_P$ ("sum of the neighbouring coefficients")

- To ensure positive coefficients (and implement non-linear schemes), advective fluxes are often decomposed into
  Upwind + deferred correction
  with the latter being transferred to the source term and treated explicitly (i.e. fixed for this iteration)
- **Boundary conditions** can be implemented by transferring boundary fluxes to the source terms
- **Under-relaxation** is usually required to solve coupled and/or non-linear equations