**Enrico Nobile, Mitja Morgut, Paola Ranut**
*Dipartimento di Ingegneria e Architettura*
*Università  degli Studi di Trieste*

*Corso di Termofluidodinamica Computazionale*

# Esercizi da svolgere
# AA 2013/2014

Marzo 2014

# Problema 1

## Le superfici alettate

L'uso di di superfici alettate per incrementare lo scambio termico fra superfici solide e fluidi trova molteplici applicazioni quali, a titolo di esempio, il raffreddamento ad aria di motori a combustione interna, gli scambiatori di calore liquido-gas ed il raffreddamento di componenti elettronici. La figura 1 riporta due esempi di superfici alettate.

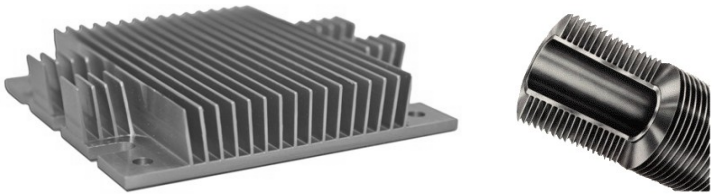

Figura 1: Esempi di superfici alettate: a sinistra dissipatore per componenti elettronici, a destra tubo alettato.

Le alette possono assumere configurazioni diverse, a seconda delle applicazioni, ed il caso più semplice è costituito da un'aletta piana a sezione rettangolare costante, come illustrato in figura 2.

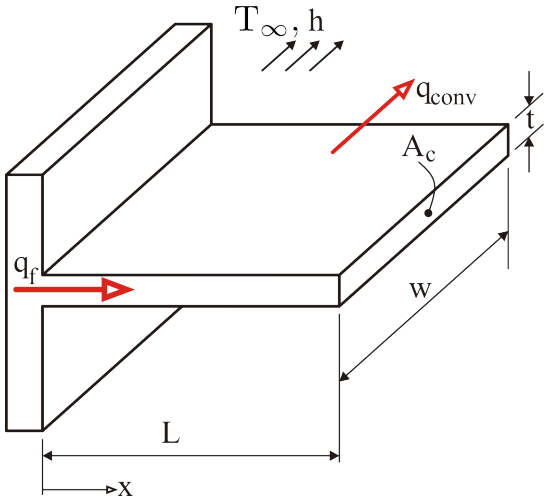


Figura 2: Aletta piana a sezione uniforme.

Per valutare le prestazioni termiche di un'aletta, e quindi dell'intera superficie alettata, è necessario determinare la distribuzione di temperatura nell'aletta stessa. Per far ciò è comune adottare alcune ipotesi semplificative:

1. Andamento *monodimensionale* del campo termico all'interno dell'aletta: $T \approx T(x)$.
2. Regime stazionario.
3. Conducibilità termica $k$ costante.

4. Scambio termico per irraggiamento trascurabile.

5. Assenza di generazione termica.

6. Coefficiente di scambio termico convettivo $h$ costante lungo tutta l'aletta.

In particolare l'ipotesi 1. è usualmente giustificata dal piccolo spessore dell'aletta rispetto alla sua lunghezza, che pertanto da luogo a variazioni di temperatura, nella sezione trasversale dell'aletta, trascurabili rispetto alle differenze di temperatura fra questa e l'ambiente.
Con tali ipotesi, ed assumendo che la sezione $A_c$ sia costante o vari secondo una legge lineare o parabolica, è possibile procedere per via analitica, ricorrendo ad un bilancio di energia per un concio elementare dell'aletta di lunghezza $\mathrm{d}x$. Operando in tal modo si perviene ad un'equazione differenziale del secondo ordine, la cui soluzione generale contiene due costanti che possono venir determinate in base alle condizioni al contorno [1, 2].

## Soluzione numerica con il metodo dei Volumi Finiti

Con il metodo dei Volumi Finiti è tuttavia possibile evitare sin dal principio questo passaggio - dal *finito* al *differenziale* e poi nuovamente al *finito* - suddividendo l'aletta in un certo numero $N$ di Volumi di Controllo, ed imponendo il bilancio di energia su ciascuno di essi, come schematizzato in figura 3 nella quale, per chiarezza di rappresentazione, si è considerata la sola proiezione sul piano.

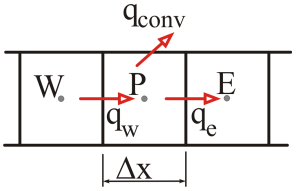


Figura 3: Bilancio su un volume di controllo dell'aletta.

Tale bilancio conduce alla

$$-q_w + q_e + q_{conv} = 0$$

ed esplicitando i flussi termici conduttivi attraverso la legge di Fourier ed il flusso termico convettivo con la legge di Newton si ha

$$kA_c \left.\frac{\mathrm{d}T}{\mathrm{d}x}\right|_w - kA_c \left.\frac{\mathrm{d}T}{\mathrm{d}x}\right|_e + P\Delta x\, h\,(T_P - T_\infty) = 0$$

dove $P$ è il perimetro dell'aletta e $A_c$ la sua sezione trasversale.
Utilizzando infine lo schema CDS del 2o ordine per la valutazione dei flussi conduttivi e raccogliendo si ottiene, per il generico volume di controllo interno (per i volumi di controllo alle estremità le espressioni possono differire)

$$A_W T_W + A_P T_P + A_E T_E = S_P$$

con

$$
\begin{aligned}
S_P &= -P\,\Delta x\,h\,T_\infty \\
A_W &= \frac{kA_c}{\Delta x} \\
A_E &= \frac{kA_c}{\Delta x} \\
A_P &= -\left(A_W + A_E\right) - P\,\Delta x\,h
\end{aligned}
$$

Introducendo le condizioni al contorno nelle espressioni per la cella 1 e la cella $N$, è facile verificare che si ottiene un sistema tridiagonale, che può essere convenientemente risolto, in MATLAB, sfruttando la funzione `spdiags`.

Il vantaggio del metodo dei Volumi Finiti - e di qualunque analogo metodo numerico - è la notevole flessibilità: è infatti possibile trattare facilmente casi di alette con sezione trasversale variabile in modo arbitrario, tener conto di proprietà termofisiche variabili e del possibile contributo dello scambio termico per irraggiamento etc. Infine nel caso non sia verificato l'andamento *monodimensionale* del campo termico all'interno dell'aletta, il problema può ancora venir affrontato ricorrendo ad una discretizzazione bi- o tridimensionale.

## Problema proposto

Un'aletta piana di sezione uniforme è costruita con un materiale avente conducibilità termica $k = 50$ W/(m K), spessore $t = 20$ mm, lunghezza $L = 200$ mm, ed è molto profonda nella direzione normale alla pagina. Il coefficiente di scambio termico convettivo vale $h$ =500 W/(m$^2$ K) con una temperatura dell'ambiente $T_\infty = 25$ °C. La base dell'aletta è mantenuta ad una temperatura $T_b = 200$ °C, mentre anche l'estremità dell'aletta scambia calore per convezione.

Assumendo una distribuzione *1D* della temperatura valutare, con il metodo dei Volumi Finiti, il flusso termico unitario (per metro di profondità) scambiato dall'aletta $q'_f$ [W/m], al variare del numero $N$ di suddivisioni, con $N$ = 5, 10, 20, 40 e 80. Riportare in grafico l'andamento dell'errore percentuale al variare di $N$, dove l'errore è valutato come differenza fra il valore del flusso $q'_f$ ottenuto dalla soluzione numerica e quello dato dalla soluzione analitica, pari a

$$
q_f = M\,\frac{\sinh mL + (h/mk)\cosh mL}{\cosh mL + (h/mk\sinh mL}
$$

con

$$
m = \sqrt{hP/kA_c}
$$

$$
M = \sqrt{hPkA_c}\,(T_b - T_\infty)
$$

# Problema 2

## La Norma UNI EN ISO 10211

La norma Europea *EN ISO 10211 - Thermal bridges in building construction - Heat flows and surface temperatures - Detailed calculations* del 2007, recepita in Italia dall'UNI (Ente Nazionale Italiano di Unificazione) come *UNI EN ISO 10211* e con il titolo *Ponti termici in edilizia - Flussi termici e temperature superficiali - Calcoli dettagliati*, definisce le specifiche dei modelli geometrici 3D e 2D di un ponte termico, ai fini del calcolo numerico di:

- Flussi termici, ai fini di determinare le dispersioni termiche totali di un edificio o di una sua parte;
- Temperature minime superficiali, ai fini di valutare il rischio di condensazione superficiale.

La norma include i limiti del modello geometrico e le sue suddivisioni, le condizioni limite ed i valori termici associati da utilizzare.

## Introduzione

La valutazione dei flussi termici (dispersioni) negli edifici, di grande rilevanza per la determinazione dei consumi e della *qualità energetica* di questi, può venir effettuata, con accuratezza sufficiente ai fini pratici, ricorrendo all'ipotesi di flusso monodimensionale (1D), per il quale, com'è noto, le soluzioni analitiche sono particolarmente semplici.
Tuttavia, negli edifici esistono *singolarità*, quali angoli, giunzioni, elementi finestrati, variazioni di composizione e/o spessore etc., tali da dar luogo a campi termici, e quindi flussi, 2D o 3D. Tali singolarità sono chiamate *ponti termici* (Ingl. *thermal bridges*), e, in confronto alle pareti piane, danno luogo a:

1. Variazioni del flusso termico;
2. Variazioni delle temperature superficiali interne

ambedue importanti.

La norma segnala che, sebbene per ambedue le variazioni le metodologie di calcolo siano simili, esse non sono necessariamente identiche per la valutazione dei flussi o delle temperature superficiali.

Come già accennato, i ponti termici danno luogo a campi termici, e flussi, bi- o tridimensionali, che possono venir calcolati accuratamente (accuratezza tipica $\pm 5\%$) solo facendo ricorso a metodologie di tipo numerico, oggetto appunto della norma in esame. Vi è da aggiungere, inoltre, che in numerose situazioni modelli numerici 2D forniscono risultati adeguati, soprattutto quando gli elementi costruttivi (es. angoli) sono uniformi in una direzione.

Infine, è doveroso segnalare che, qualora l'accuratezza necessaria sia inferiore, è possibile evitare l'uso di modelli numerici, ed utilizzare viceversa metodologie semplificate, quali *cataloghi* di ponti termici (accuratezza dell'ordine del $\pm 20\%$), calcoli manuali (accuratezza

$\pm 20\%$) e utilizzo di valori standard (accuratezza $\pm 50\%$). La descrizione e l'utilizzo di tali modelli semplificati è discusso nella norma UNI *UNI EN ISO 14683 - Ponti termici in edilizia - Coefficienti di trasmissione termica lineica - Metodi semplificati e valori di riferimento*

## Validazione dei metodi di calcolo

Al fine di poter classificare un metodo di calcolo numerico come *Metodo di elevata precisione per calcoli stazionari 3D*, la norma, nell'*Annex A - Validation of calculation methods*, descrive quattro casi (test reference cases 1, 2, 3 e 4) da utilizzare per la validazione.

Analogamente, la norma specifica il subset di casi precedenti (casi 1 e 2) per la validazione necessaria a classificare un metodo come *Metodo di elevata precisione per calcoli stazionari 2D*

Com'è ovvio, i casi 1 e 2 sono bi-dimensionali, mentre i casi 3 e 4 sono tri-dimensionali

## Case 1

Il Case 1, certamente il più semplice, è particolarmente interessante, poiché ammette una soluzione analitica. Esso è relativo allo scambio termico attraverso mezza (per ragioni di simmetria) colonna quadrata, per la quale siano note le temperature ai bordi. La norma, in particolare, si limita ad indicare, come illustrato in figura 4, il valore della soluzione analitica su una griglia equidistante di 28 punti, riportata in tabella 1, ed impone che la differenza fra i valori calcolati e forniti delle temperature non ecceda $0.1$ °C.

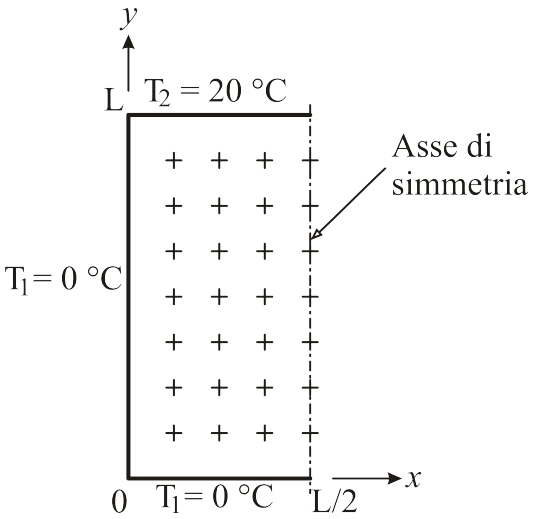


Figura 4: UNI EN ISO 10211: Reference Case 1

Per tale problema - scambio termico conduttivo stazionario in assenza di generazione interna - l'equazione generale di trasporto si semplifica nella:

$$\nabla \cdot (\lambda \nabla t) = 0$$

| Temperature (°C) | | | |
|---|---|---|---|
| 9.7 | 13.4 | 14.7 | 15.1 |
| 5.3 | 8.6 | 10.3 | 10.8 |
| 3.2 | 5.6 | 7.0 | 7.5 |
| 2.0 | 3.6 | 4.7 | 5.0 |
| 1.3 | 2.3 | 3.0 | 3.2 |
| 0.7 | 1.4 | 1.8 | 1.9 |
| 0.3 | 0.6 | 0.8 | 0.9 |

Tabella 1: Valori di temperatura nei punti equidistanti della griglia di figura 1 ottenuti dalla soluzione analitica.

e tenendo conto che operiamo in 2D con proprietà termofisiche costanti, la precedente può esplicitarsi nella:

$$\frac{\partial^2 t}{\partial x^2} + \frac{\partial^2 t}{\partial y^2} = 0$$

La soluzione analitica, non riportata dalla Norma, è tuttavia facilmente reperibile in letteratura [2], ed è data, in forma adimensionale per l'intera colonna quadrata di lato $L$, dalla:

$$\theta\left(x,y\right) = \frac{2}{\pi}\sum_{n=1}^{\infty}\frac{\left(-1\right)^{n+1}+1}{n}\sin\frac{n\pi x}{L}\frac{\sinh\left(n\pi y/L\right)}{\sinh\left(n\pi\right)}$$

dove $\theta$ è la temperatura adimensionale:

$$\theta \equiv \frac{T-T_1}{T_2-T_1}$$

L'andamento del campo termico, ottenuto numericamente, è riportato, a scopo illustrativo, in figura 5.

## Quesiti

1. Sviluppare, in MATLAB, SCILAB o Python, una procedura numerica per la soluzione del problema, basata sul metodo dei volumi finiti (FV), utilizzando una griglia Cartesiana ortogonale 2D a passo costante in $x$ ed $y$, tenendo conto che, in generale, $\Delta x \neq \Delta y$.

2. Valutare il numero minimo di celle (volumi finiti), per ciascuna delle due direzioni $x$ ed $y$, sufficienti a garantire l'accuratezza minima richiesta dalla norma.

3. Per almeno quattro griglie - costruite raddoppiando, partendo dalla griglia più rada, il numero di celle in ciascuna direzione $x$ ed $y$ - valutare e riportare in forma tabellare e grafica, il tempo di calcolo necessario, utilizzando le seguenti metodologie di soluzione del sistema di equazioni:

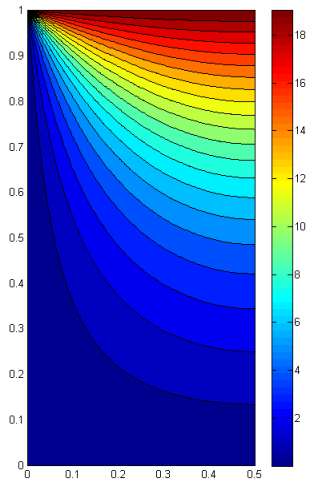


Figura 5: Campo termico per il Reference Case 1 della UNI EN ISO 10211.

(a) Metodo diretto (divisione a sinistra *backslash* in MATLAB);

(b) Metodo iterativo basato sul gradiente coniugato, eventualmente precondizionato (in MATLAB, sono disponibili numerose alternative);

(c) Metodo SOR, verificando l'influenza del parametro $\omega$ di sovrarilassamento.

(d) Metodo di Jacobi.

## Interpolazione Bilineare

Per la determinazione delle temperatura nei punti indicati dalla norma, che in generale *non* coincidono con i centroidi delle celle nei quali sono disposte le variabili, è possibile far ricorso, trattandosi di un mezzo omogeneo ed isotropo, alla cosiddetta *Interpolazione Bilineare*, descritta nel seguito.

In matematica, l'*Interpolazione Bilineare* rappresenta la naturale estensione dell'*Interpolazione Lineare* per interpolare funzioni di due variabili su una griglia regolare. Essa trova ampio impiego, ad esempio, nell'elaborazione digitale dell'immagine, e costituisce la metodologia fondamentale per il ridimensionamento di immagini. Essa, in tale ambito, è anche chiamata *Bilinear Filtering* o *Bilinear Texture Mapping*.

Con riferimento alla figura 6, si supponga di voler determinare il valore della temperatura $T$ nel punto $P(x, y)$, assumendo noti i valori di temperatura nei quattro punti $Q_{11} = (x_1, y_1)$, $Q_{12} = (x_1, y_2)$, $Q_{21} = (x_2, y_1)$ e $Q_{22} = (x_2, y_2)$.

Si può pensare di effettuare dapprima un'interpolazione lineare lungo $x$, ottenendo:

$$T(R_1) \approx \frac{x_2 - x}{x_2 - x_1} T(Q_{11}) + \frac{x - x_1}{x_2 - x_1} T(Q_{21})$$

dove $R_1 = (x, y_1)$,

$$T(R_2) \approx \frac{x_2 - x}{x_2 - x_1} T(Q_{12}) + \frac{x - x_1}{x_2 - x_1} T(Q_{22})$$

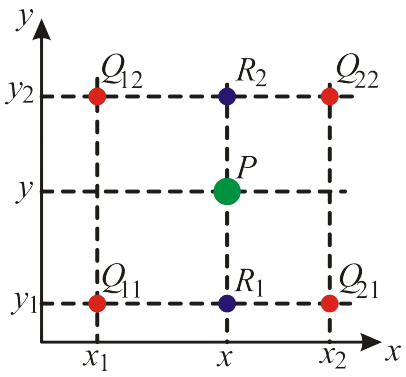


Figura 6: Interpolazione Bilineare:

dove $R_2 = (x, y_2)$.

Si procede poi interpolando linearmente in direzione $y$:

$$T(P) \approx \frac{y_2 - y}{y_2 - y_1} T(R_1) + \frac{y - y_1}{y_2 - y_1} T(R_2)$$

Sostituendo nell'ultima espressione le relazioni precedenti, otteniamo la stima di $T(x, y)$:

$$\begin{aligned} T(x,y) \approx & \frac{T(Q_{11})}{(x_2 - x_1)(y_2 - y_1)}(x_2 - x)(y_2 - y) \\ & + \frac{T(Q_{21})}{(x_2 - x_1)(y_2 - y_1)}(x - x_1)(y_2 - y) \\ & + \frac{T(Q_{12})}{(x_2 - x_1)(y_2 - y_1)}(x_2 - x)(y - y_1) \\ & + \frac{T(Q_{22})}{(x_2 - x_1)(y_2 - y_1)}(x - x_1)(y - y_1) \end{aligned}$$

È forse utile far notare che:

1. A dispetto del nome, l'interpolante *non* è lineare: in particolare l'interpolante è lineare per linee parallele all'asse $x$ o all'asse $y$, mentre per altre direzioni è *quadratico*.
2. Il risultato dell'interpolazione bilineare è indipendente dall'ordine delle interpolazioni.
3. L'ovvia estensione dell'interpolazione bilineare al caso 3D è chiamata *Interpolazione Trilineare*.

# Problema 3

## L'equazione di Burgers

Si consideri l'equazione di Burgers monodimensionale (1D), espressa dalla:

$$\frac{\partial u}{\partial \vartheta} + u\frac{\partial u}{\partial x} = \nu\frac{\partial^2 u}{\partial x^2}$$

in forma non-conservativa, e dalla:

$$\frac{\partial u}{\partial \vartheta} + \frac{1}{2}\frac{\partial}{\partial x}\left(u^2\right) = \nu\frac{\partial^2 u}{\partial x^2}$$

in forma conservativa.

L'equazione di Burgers, così chiamata in onore dello scienziato Olandese J.M. Burgers [3], è di notevole importanza nello studio di svariati problemi dell'ingegneria, quali, ad esempio. flussi con onde d'urto, propagazione non-lineare delle onde, turbolenza. Essa può essere considerata una versione semplificata delle equazioni di Navier-Stokes, nella quale, pur essendo presente il termine viscoso ed il termine avvettivo non-lineare, manca il gradiente di pressione e pertanto può venire applicata e studiata a problemi 1D, con ovvi vantaggi in termini di economia di calcolo e la disponibilità - molto rara per problemi non-lineari - di soluzioni in forma analitica.

## Problema proposto

Si consideri il problema nel quale le condizioni al contorno, nell'intervallo $0 \leq x \leq 1$, siano le seguenti:

$$u = 1 \;\text{ a }\; x = 0 \quad \text{e} \quad u = 0 \;\text{ a }\; x = 1$$

La soluzione analitica, per queste condizioni al contorno [4], è data dalla

$$u = u_R + \frac{1}{2}\left(u_L - u_R\right)\left[1 - \tanh\left(\frac{\left(u_L - u_R\right)\left(x - V_S\,\vartheta\right)}{4\nu}\right)\right]$$

e rappresenta bene il transito di un onda d'urto - *shock* - in un fluido viscoso, dove

$$V_S = \frac{u_L + u_R}{2}$$

rappresenta la sua velocità di propagazione, e $u_L$ e $u_R$, definite

$$u_L = \lim_{x\to-\infty} u, \quad u_R = \lim_{x\to\infty} u, \quad u_L > u_R$$

possono venire assunte pari ai valori di $u$ al contorno.

Assumendo pertanto, come condizione iniziale, la soluzione all'istante $\vartheta = 0.2$, si confronti la soluzione analitica con quella ottenuta numericamente, usando il metodo dei Volumi Finiti, considerando lo schema CDS per la discretizzazione spaziale - per ambedue i contributi avvettivo e diffusivo - ed i metodi di Eulero impliciti del primo e second'ordine per l'integrazione temporale. In particolare, si confronti la soluzione numerica con quella analitica, assumendo $\nu = 1 \times 10^{-3}$, per $\vartheta = 0.7, 1.5$, e si considerino i seguenti casi:

1. $N = 250$, $\Delta\vartheta = 0.01$;
2. $N = 500$, $\Delta\vartheta = 0.01$;
3. $N = 1000$, $\Delta\vartheta = 0.01$;
4. $N = 250$, $\Delta\vartheta = 0.001$;
5. $N = 500$, $\Delta\vartheta = 0.001$;
6. $N = 1000$, $\Delta\vartheta = 0.001$;

dove $N$ è il numero di volumi nella discretizzazione spaziale, e $\Delta\vartheta$ è il time-step.

Osservando infine che, utilizzando uno schema di integrazione implicito per un'equazione non-lineare è spesso opportuno (o necessario) procedere a sub-iterazioni per time-step, si valuti anche l'effetto di queste sull'accuratezza, considerando, ad esempio, $1$ (nessuna sub-iterazione), $5$ e $10$ sub-iterazioni.

Si riportino i risultati in forma sintetica, grafica e/o tabellare.

# Problema 4

Sviluppare, in MATLAB o altro linguaggio a scelta, una procedura di calcolo - script o function - in grado di:

- Leggere una griglia generica 3D secondo il formato di *OpenFOAM* [5].
- Rappresentare graficamente la griglia di superficie, composta da tutte le *patches* che ne costituiscono le superfici di contorno.
- Rappresentare graficamente, nella medesima figura del punto precedente, le normali a ciascuna faccia della griglia di superficie (boundaries), verificando che essa sia rivolta verso l'esterno del dominio.
- Valutare la superficie esterna totale della griglia di calcolo.
- Calcolare il volume totale racchiuso dalla griglia.

## Formato dei files

Sulla pagina *Moodle* del corso sono fornite le griglie per due geometrie arbitrarie, `geom1.zip` e `geom2.zip`. Rimandando alla *Programmer's Guide* di *OpenFOAM* per ulteriori informazioni [5], la griglia, in OpenFOAM, è descritta dai seguenti files:

**points** Contiene la lista, la cui numerazione implicita parte da $0$, delle coordinate $x, y, z$ di tutti i nodi (vertici) della griglia. Il formato è il seguente:

```
/*---------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.1.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       vectorField;
    location    "constant/polyMesh";
    object      points;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //


7672
(
(0 0 -25)
(100 3.061515885e-15 -425)
(150 0 -425)
......
......
(87.86430881 22.96002865 -37.13895599)
(88.56119186 21.05934167 -24.30295503)
)


// ************************************************************************* //
```

**faces** Contiene la lista, anche qua con numerazione implicita che parte da 0, di tutte le facce costituenti la griglia. Poiché in OpenFOAM le celle sono costituite da poliedri arbitrari, le cui facce sono anch'esse poligoni con un numero arbitrario di vertici/lati, per ogni faccia vengono indicati il numero di nodi ed i nodi (vertici) descritti nel file **points**. Il formato è il seguente:

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.1.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       faceList;
    location    "constant/polyMesh";
    object      faces;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //


47329
(
3(3887 1662 1708)
3(3887 1708 2231)
3(1662 2231 1708)
................
................
4(117 211 1774 1776)
4(208 117 1776 1775)
)


// ************************************************************************* //
```

È importante osservare che:

- La sequenza dei nodi per ciascuna faccia non è arbitraria, ma segue alcune regole precise:
  1. Ciascun vertice, nell'elenco, è collegato con un *edge* ai due nodi adiacenti nella lista.
  2. Il verso di numerazione dei nodi, nell'elenco, definisce il verso della normale alla faccia attraverso la nota regola della mano destra (o del cavatappi), come indicato in figura 7.

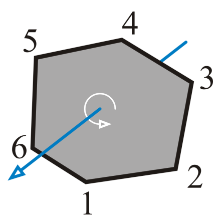


Figura 7: Orientamento della faccia in OpenFOAM.

- Vi sono due tipi di facce:

  **Internal faces** Si tratta di facce che connettono due - e non potrebbero essere più di due - celle. Per ciascuna di questa facce *interne*, l'ordinamento dei vertici è tale da garantire che la normale alla faccia *punti* in direzione della cella avente indice maggiore: ad esempio, per la faccia che connette le celle 27 e 54, la normale è rivolta verso la cella 54.

  **Boundary faces** Si tratta delle facce che appartengono ad una sola cella, poiché giacciono sul contorno. Perciò una *boundary face* è collegata ad una sola cella ed ad una *boundary patch*. L'ordinamento dei vertici è tale da assicurare che la normale alla faccia è rivolta all'esterno del dominio di calcolo.

  *Sebbene tale situazione non rientri nei due problemi assegnati, si segnala, per completezza, che vi possono essere casi nei quali una o più facce siano costituite da più di 10 vertici. In tale caso, la modalità di scrittura dei nodi della faccia cambia da una di tipo* orizzontale *ad una di tipo* verticale *[6]. In particolare viene riportata dapprima una linea vuota (blank line), nella riga successiva una parentesi tonda aperta, ed a seguire un numero di righe, pari al numero di vertici, ognuna riportante l'indice (label) del vertice stesso. A finire una riga con parentesi tonda chiusa ed una blank line. Un esempio per una faccia con 11 nodi è riportato nel seguito:*

```
4(78878 221293 235727 221792)
4(99528 222236 236316 222495)

11
(
221792
235727
219017
15669
103938
170589
116587
222495
236316
222236
78878
)

4(15670 219019 236319 157010)
4(236319 210336 205052 157010)
```

**boundary** Contiene l'elenco delle *patches* che costituiscono le regioni del contorno, sulle quali vengono applicate le opportune condizioni al contorno. Il numero delle *patches* è arbitrario, e ogni *patch* viene identificata da un nome, dal tipo, dal numero di facce che la costituisce ed infine dalla faccia iniziale, tenendo presente che:

- la numerazione implicita delle facce, nel file **faces**, è tale da assicurare che ogni patch è costituita da una lista di facce contigue;
- la numerazione della facce, nel file **faces**, è tale da garantire che le boundary patches siano costituite da blocchi contigui di facce e che inoltre essi siano costituiti dalle facce con numerazione maggiore.

Il formato è il seguente:

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.1.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       polyBoundaryMesh;
    location    "constant/polyMesh";
    object      boundary;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

3
(
    INGRESSO
    {
        type            wall;
        nFaces          86;
        startFace       43878;
    }
    PARETE
    {
        type            wall;
        nFaces          3285;
        startFace       43964;
    }
    USCITA
    {
        type            wall;
        nFaces          80;
        startFace       47249;
    }
)

// ************************************************************************* //
```

**owner** Contiene, sempre con numerazione implicita da 0, la lista delle celle designate come *owner* per ogni faccia, interna ed esterna (contorno), descritta dal file **faces**. Poiché *tutte* le facce, interne ed esterne, possiedono una cella *owner*, il numero totale di *entries* del file **owner** coincide con il numero di facce nel file **faces**. Il formato è il seguente:

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.1.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       labelList;
    note        "nPoints:7672  nCells:20338  nFaces:47329  nInternalFaces:43878";
    location    "constant/polyMesh";
    object      owner;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
```

```
47329
(
0
0
0
.
.
17258
17254
)


// ************************************************************************* //
```

**neighbour** Contiene la lista delle celle designate come *neighbour* per ogni faccia interna, poiché ogni faccia esterna ha solo un *owner* e non un *neighbour*. Sulla base di quanto detto al riguardo del file **boundary**, il numero di *entries* del file **neighbour** è pari al numero totale di facce meno il numero delle facce costituenti il contorno. Il formato è il seguente:

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.1.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       labelList;
    note        "nPoints:7672  nCells:20338  nFaces:47329  nInternalFaces:43878";
    location    "constant/polyMesh";
    object      neighbour;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //


43878
(
152
674
1003
.
.
20335
20337
)


// ************************************************************************* //
```

È infine forse utile notare che ambedue i files **owner** e **neighbour** riportano, nelle righe di intestazione, il numero di nodi `nPoints`, di celle `nCells`, di facce `nFaces` e di facce interne `nInternalFaces`.

## Metodologia

Per poter procedere alla valutazione del volume racchiuso dalla griglia di calcolo, è conveniente, piuttosto che sommare il volume di tutte le celle, utilizzare l'espressione, vista a lezione e

riportata sul testo, eq. (2.142), che fa uso del teorema di Gauss, qua riportata per completezza. Infatti, utilizzando l'identità:

$$\nabla \mathbf{r} = \nabla \left( x\mathbf{i} + y\mathbf{j} + z\mathbf{k} \right) \equiv 3$$

il volume racchiuso dalla griglia di calcolo si può calcolare sfruttando il teorema di Gauss:

$$V = \frac{1}{3} \int_V \nabla \mathbf{r} \, \mathrm{d}V = \frac{1}{3} \int_A \mathbf{r} \cdot \mathbf{n} \mathrm{d}A = \frac{1}{3} \sum_{i=1}^{N_{ef}} \mathbf{r}_i \cdot \mathbf{n}_i \, A_i$$

In questo caso il volume $V$ di integrazione è costituito dall'intero dominio di calcolo, la superficie di integrazione $A$ è data dalla superficie esterna della griglia e $N_{ef}$ rappresenta il numero di facce della griglia disposte sul contorno.

Per poter quindi calcolare il volume $V$ e l'area $A$ è necessario valutare:

1. L'area delle facce della griglia che giacciono sul contorno, eq. (2.135).
2. Il centroide delle facce della griglia che giacciono sul contorno, eq. (2.136) e (2.137).

### Suggerimenti ed osservazioni

È preferibile, in questo caso, procedere per passi, sviluppando lo *script*, e le associate *function*, per gradi, partendo dalla lettura dei files e procedendo poi alla rappresentazione grafica delle mesh di superficie etc.
Si tenga inoltre presente che:

- In MATLAB arrays e vettori sono indicizzati da $1$ e non da $0$;
- In MATLAB esistono numerosi comandi a basso livello per poter manipolare stringhe di caratteri ed estrarne i campi (nomi, numeri etc.);
- La funzione di MATLAB `patch` consente di rappresentare graficamente poligoni/facce 3D arbitrarie provvedendo inoltre alla rimozione automatica delle linee nascoste.
- Per eseguire prodotti scalari e vettoriali fra vettori, in MATLAB è comodo utilizzare, rispettivamente, le funzioni `dot` e `cross`.

## Riferimenti bibliografici

[1] G. Comini, G. Cortella, *Fondamenti di trasmissione del calore*, 4a Ed., S.G.E. Editore, (2013).

[2] F. P. Incropera, D. P. Dewitt, T. L. Bergman, A. S. Lavine, *Fundamentals of Heat and Mass Transfer*, 6th Ed., Wiley, (2007).

[3] J. M. Burgers, A mathematical model illustrating the theory of turbulence. *Advances in Applied Mechanics*, Vol. 1, Academic Press, New York, pp. 171-199, (1948).

[4] K. Masatsuka, I do like CFD, `http://www.cfdbooks.com/`, 2009.

[5] AA. VV., OpenFOAM Programmer's Guide, `http://www.openfoam.org/docs/`.

[6] N. Del Puppo, Comunicazione Privata, 2013.