POLITECNICO DI TORINO

Collegio di Ingegneria Chimica e dei Materiali

Corso di Laurea Magistrale
in Ingegneria Chimica e dei Processi Sostenibili

Tesi di Laurea Magistrale

Reattori multifase: modellazione tramite CFD e Machine Learning

Relatori

Prof. Gianluca BOCCARDO

Prof. Daniele MARCHISIO

Dott.ssa Agnese MARCATO

Dott. Sandro MALUSÀ

Candidato

Enrico Salvatore ITALIANO

Anno Accademico 2023/2024

Indice

Elenco delle figure IV

Elenco delle tabelle VI

1 Introduzione 1
- 1.1 Cambiamento climatico . . . 1
- 1.2 Power-to-gas . . . 2
- 1.3 BHM . . . 3
  - 1.3.1 Principio di funzionamento . . . 4
  - 1.3.2 Principali configurazioni reattoristiche . . . 6
  - 1.3.3 Simulazioni CFD su TBR per BHM . . . 9

2 Modelli Teorici 11
- 2.1 Modello fisico . . . 11
  - 2.1.1 Ipotesi adottate . . . 11
  - 2.1.2 Equazioni di governo . . . 12
- 2.2 Modelli data-driven . . . 15
  - 2.2.1 Normalizzazione delle feature . . . 16
  - 2.2.2 Modelli di regressione . . . 17
  - 2.2.3 Reti neurali . . . 18

3 Metodi Computazionali 23
- 3.1 Metodo ai volumi finiti . . . 23
  - 3.1.1 Equazioni di governo . . . 24
  - 3.1.2 Costruzione del sistema . . . 25
  - 3.1.3 Algoritmo di risoluzione . . . 27
  - 3.1.4 Condizioni di bordo . . . 28
- 3.2 Meshing . . . 28
- 3.3 Discretizzazione spaziale . . . 28
  - 3.3.1 Termine diffusivo . . . 29
  - 3.3.2 Termine convettivo . . . 31

3.4 Discretizzazione temporale 33
3.5 OpenFOAM 35
3.5.1 Impostazione dei *foamCases* 36
3.5.2 Meshing tools 37
3.5.3 Solver 37
3.6 Risorse computazionali 37

4 Setup delle simulazioni 39
4.1 Geometria 39
4.2 Condizioni al contorno e simulazioni effettuate 40
4.2.1 Criterio di stop simulazione 41
4.2.2 Impostazione dataset 42
4.3 PostProcessing 43

5 Risultati 45
5.1 Simulazioni di CFD 45
5.2 Modello data-driven 48

6 Conclusioni 51

Elenco delle figure

1.1 Emissioni nette di GHG di origine antropica, 1990-2022 [2] 1
1.2 Schema del funzionamento di un sistema power-to-gas [3]. 3
1.3 Schema di un impianto di biometanazione in-situ [9]. 5
1.4 Schema di un impianto di biometanazione ex-situ [9]. 5
1.5 Schema di un CSTR per biometanazione [11]. 6
1.6 Schema di un BCR per biometanazione [13]. 7
1.7 Schema di un PFR per biometanazione [14]. 8
1.8 Schema di un TBR per biometanazione [15]. 8

2.1 Esempio di una curva di permeabilità relativa [19]. 14
2.2 Schema di una threshold logic unit [23]. 19
2.3 Principali funzioni di attivazione. 20
2.4 Schema di un percettrone [23]. 20
2.5 Schema di un esempio di rete MLP in cui sono presenti due hidden layer [23]. 21

3.1 Schema del processo di discretizzazione [24]. 24
3.2 Flusso attraverso le facce f della cella C [24]. 26
3.3 Topologia delle celle [24]. 29
3.4 Esempi di elementi presenti in mesh tridimensionali o bidimensionali [24]. 29
3.5 Griglia cartesiana [24]. 30
3.6 Profilo di interpolazione lineare [24]. 31
3.7 Griglia cartesiana nel problema di advezione-diffusione [24]. 31
3.8 Profilo dello schema upwind [24]. 33
3.9 Profilo dello schema CDS (schema del secondo ordine) [24]. 34
3.10 Profilo dello schema *second order upwind* (schema del secondo ordine) [24]. 34
3.11 Profilo dello schema QUICK (schema del terzo ordine) [24]. 35
3.12 Struttura di un *foamCase*. 36
3.13 Schema dell'algoritmo PIMPLE [27]. 38

4.1 Esempio di geometria monodispersa periodica randomica. 40
4.2 Distribuzione dei fluidi in un caso inizializzato con $\alpha_{water} = 0.1$. . . 41
4.3 Esempio di simulazione che ha soddisfatto il criterio di stop per un intervallo Δt pari a 1,5 s. 42

5.1 Distribuzione delle fasi per diversi valori di α_{water}. 46
5.2 Campi di velocità (componente x) per diversi valori di α_{water}. . . . 47
5.3 Permeabilità relative delle due fasi al variare di α_{water}. In rosso è rappresentata la permeabilità dell'aria κ_a, mentre in blu è mostrata quella dell'acqua κ_w. 48
5.4 *Loss curves* dell'allenamento dei modelli. 49
5.5 Diagrammi di parità della permeabilità relativa delle due fasi. . . . 49

Elenco delle tabelle

1.1 Condizioni operative dei processi di metanazione e BHM. 4

4.1 Proprietà dei fluidi simulati 40
4.2 Range di variazione dei parametri per la creazione del dataset. . . . 43

5.1 Parametri di allenamento delle reti neurali. 48

Capitolo 1

Introduzione

1.1 Cambiamento climatico

La lotta al cambiamento climatico rappresenta una delle sfide più urgenti e complesse del mondo contemporaneo. L'eccessiva emissione nell'atmosfera terrestre dei gas serra (GHG), dovuta ad una incontrollata intensificazione delle attività umane, ha portato ad un riscaldamento globale significativo. L'IPCC (*Intergovernmental Panel on Climate Change*) definisce il riscaldamento globale come "l'aumento stimato della temperatura media della superficie globale (GMST) mediato su un periodo di 30 anni, espresso relativamente ai livelli pre-industriali"[1].

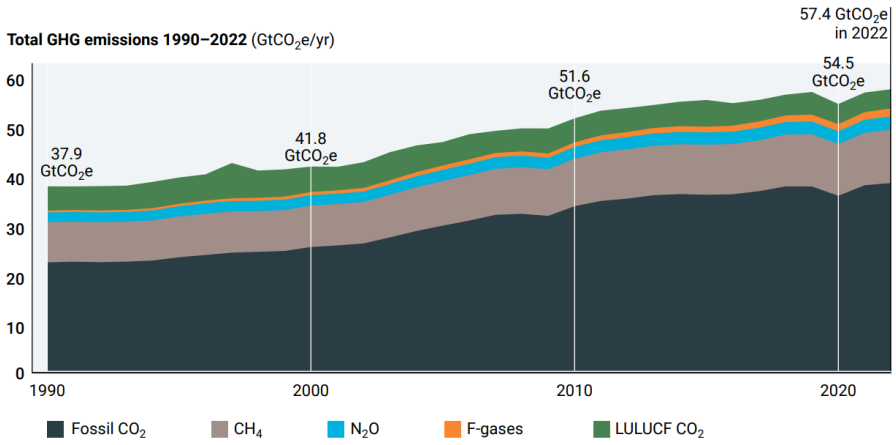


Figura 1.1: Emissioni nette di GHG di origine antropica, 1990-2022 [2]

Tale aumento è la causa di alcuni fenomeni atmosferici come l'innalzamento del livello degli oceani e l'aumentare dei fenomeni estremi legati al ciclo dell'acqua (per esempio: alluvioni, siccità e scioglimento dei ghiacciai). La comunità scientifica

concorda nell'attribuire le cause del cambiamento climatico alle emissioni di gas serra di origine antropica, responsabili dell'effetto serra. Tra i gas serra più impattanti si possono elencare il vapore acqueo (H_2O), l'anidride carbonica (CO_2), il monossido di diazoto (N_2O), il metano (CH_4), l'ozono (O_3) e i cosiddetti *F-gases* (suddivisi in HFC, PFC, SF6, tre classi diverse di gas refrigeranti). Per far fronte al cambiamento climatico, è stata stipulata nel 1992 la *Convenzione quadro delle Nazioni Unite sui cambiamenti climatici* (UNFCCC), un trattato internazionale ambientale che punta alla riduzione delle emissioni dei gas serra. Ogni anno le parti firmatarie dell'UNFCCC organizzano degli incontri formali, noti come Conferenza delle Parti (COP), in cui si valuta il progresso degli effetti delle politiche attuate per contrastare il cambiamento climatico. Uno degli obiettivi più importanti che sono stati discussi durante le varie conferenze delle parti è stato definito durante la COP 21, tenutasi a Parigi negli ultimi mesi del 2015. Tale obiettivo consiste nell'obbligo dei paesi firmatari nel mantenere l'aumento della temperatura media mondiale ben al di sotto di 2°C rispetto ai livelli preindustriali; l'accordo, inoltre, prevede un impegno attivo degli stati firmatari nell'adottare tecnologie innovative utili a limitare tale aumento a 1,5°C rispetto ai livelli preindustriali. Le varie tecnologie studiate negli anni successivi hanno tutte lo scopo di ridurre le emissioni di CO_2 o quello di raggiungere l'indipendenza dai combustibili fossili. Una tra le possibili soluzioni consiste nell'implementazione dei sistemi chiamati *power-to-gas*, che permettono di utilizzare le fonti di energia rinnovabili in maniera più efficiente e affidabile.

1.2 Power-to-gas

Uno dei più grandi limiti delle fonti di energia rinnovabili è l'instabilità della potenza prodotta; tecnologie come il fotovoltaico e l'eolico, ad esempio, dipendono molto dalle condizioni atmosferiche, dalla collocazione geografica e dal periodo dell'anno. Non essendo fonti di energia che producono una potenza costante, dunque, il loro impiego porterebbe a delle oscillazioni tra periodi in cui si dispone di un surplus di elettricità rispetto alla domanda dell'utenza e altri in cui la domanda supera la potenza disponibile. Uno metodi che si possono impiegare per ridurre l'instabilità di queste tecnologie è l'utilizzo di sistemi *power-to-gas*.
Il power-to-gas (PtG o P2G) è una tecnologia che sfrutta l'energia elettrica per produrre combustibili gassosi. La maggior parte dei sistemi PtG sfrutta l'*elettrolisi* dell'acqua per produrre idrogeno; è possibile distinguere due categorie di sistemi in base all'utilizzo dell'idrogeno prodotto:

– *single-stage PtG systems*: l'idrogeno prodotto dall'elettrolisi viene utilizzato direttamente come fonte di energia;

– *double-stage PtG systems*: l'idrogeno viene convertito in combustibili come metano, syngas o GPL.

Il sistema PtG, tramite la produzione di un gas, fornisce un metodo per immagazzinare energia durante i periodi di surplus di potenza generata dalle fonti rinnovabili; la stessa energia può essere recuperata e utilizzata durante le fasi di deficit di potenza tramite un impianto di cogenerazione. In Fig.(1.2) è mostrato il funzionamento di un sistema PtG affiancato ad un impianto che genera potenza elettrica variabile nel tempo. In questo capitolo si porrà l'attenzione sui double-stage PtG systems,

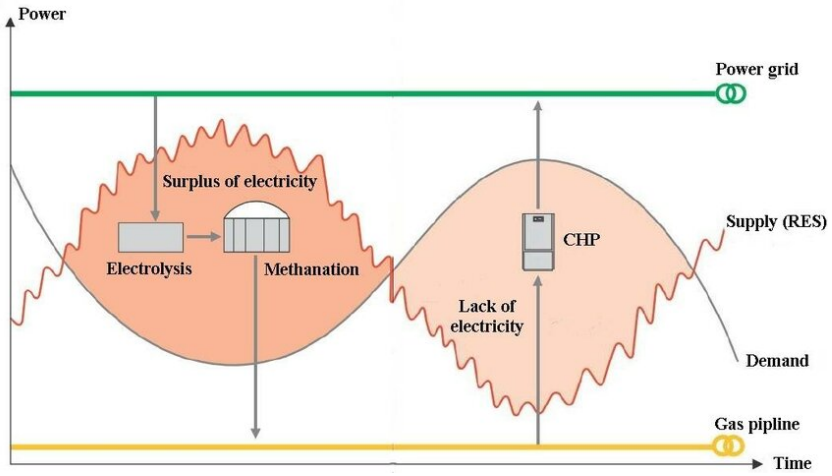


Figura 1.2: Schema del funzionamento di un sistema power-to-gas [3].

in particolare sui sistemi che sfruttano l'accoppiamento dell'elettrolisi dell'acqua alla *biometanazione* (BHM), convertendo l'idrogeno in *biometano*.

1.3 BHM

La biometanazione (*biological hydrogen methanation*, BHM) è un processo che consente di convertire l'anidride carbonica (CO_2) in biometano (CH_4); la reazione che avviene è la seguente:

$$CO_2 + 4\,H_2 \longrightarrow CH_4 + 2\,H_2O \qquad \Delta H^\circ = -164 \text{ kJ/mol} \tag{1.1}$$

La reazione è esotermica ed è la stessa che avviene nella metanazione "classica", chiamata processo Sabatier; questo processo è ampiamente utilizzato, tra le varie applicazioni, per rimuovere gli eccessi di CO_2 a valle di un impianto di steam reforming [4]. Il processo Sabatier avviene in un reattore adiabatico caratterizzato da un letto impaccato di catalizzatore al nichel che opera ad alte temperature e

ad alta pressione. Il processo di biometanazione, invece, viene condotto a temperature e pressioni inferiori e il catalizzatore è costituito da dei *batteri idrogenotrofi metanogeni*; questo tipo di batteri appartiene al dominio degli *Archea* ed è in grado di utilizzare l'idrogeno come fonte di energia, producendo metano.

	Metanazione	**Biometanazione**
Temperatura	400 °C	0 ÷ 122 °C [5]
Pressione	30 bar	1 ÷ 10 bar [6]
Catalizzatore	Nichel	Batteri metanogeni

Tabella 1.1: Condizioni operative dei processi di metanazione e BHM.

Nella tabella 1.1 sono riassunte le differenze tra le condizioni operative dei due diversi modi di condurre la reazione 1.1. Rispetto al processo Sabatier, la BHM richiede condizioni operative più facili da raggiungere, portando ad avere dei costi operativi minori [7]; inoltre, risulta essere un processo molto meno sensibile alle impurità dei gas da trattare (composti come l'H_2S e l'NH_3 non avvelenano il catalizzatore, bensì possono essere sfruttati come nutrienti per i microrganismi metanogeni). Infine, un altro aspetto vantaggioso del processo BHM è rappresentato dalla possibilità di rigenerare il catalizzatore in maniera continua, grazie alla crescita batterica [6]. Per questi motivi, il processo di BHM è diventato un importante oggetto di studio per la comunità scientifica.

1.3.1 Principio di funzionamento

Come descritto nelle sezioni precedenti, si può sfruttare la biometanazione per progettare un sistema power-to-gas: l'idrogeno necessario viene prodotto dall'elettrolisi dell'acqua, alimentata da fonti di energia rinnovabili durante i periodi di surplus rispetto alla domanda dell'utenza; l'anidride carbonica, invece, può essere fornita da qualsiasi processo industriale che produce come scarto. Tuttavia, la maggior parte delle applicazioni vedono l'accoppiamento della BHM con un impianto di digestione anaerobica.
La digestione anaerobica (AD) è un processo in cui del materiale organico viene degradato, in assenza di ossigeno, da dei microrganismi. Il prodotto di questo processo è chiamato *biogas*: una miscela di metano e anidride carbonica di composizione variabile (può contenere anche piccole quantità di acqua e acido solfidrico). Il biogas è considerato un vettore energetico rinnovabile [8]; l'integrazione della BHM a valle di un impianto di digestione anaerobica permette di aumentarne la purezza, abbattendo inoltre le emissioni di anidride carbonica. In Fig.(1.3) e Fig.(1.4) sono raffigurate le due principali configurazioni progettate per accoppiare la biometanazione alla digestione anaerobica:

– biometanazione *in-situ*: la reazione 1.1 avviene direttamente all'interno dell'impianto di digestione anaerobica. L'idrogeno proveniente dall'elettrolisi viene introdotto nel bioreattore in cui sono presenti sia i batteri responsabili della produzione di biogas sia quelli che catalizzano la biometanazione;

– biometanazione *ex-situ*: la reazione 1.1 avviene in un impianto a sé stante. Il reattore di biometanazione è posto vicino all'impianto di digestione anaerobica, che fornisce l'anidride carbonica necessaria a condurre la reazione.

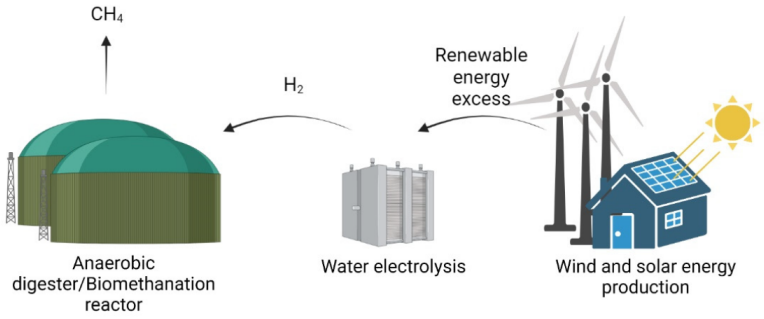


Figura 1.3: Schema di un impianto di biometanazione in-situ [9].

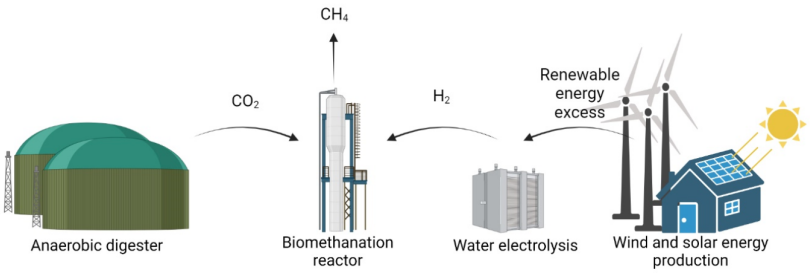


Figura 1.4: Schema di un impianto di biometanazione ex-situ [9].

Sono state studiate anche delle configurazioni ibride che prevedono un collegamento in serie tra un sistema in-situ e uno ex-situ [10]. Entrambe le configurazioni descritte in precedenza presentano dei pro e dei contro: il più grande vantaggio dell'impianto in-situ è il minor costo di installazione; tuttavia, questa opzione richiede che l'idrogeno alimentato sia costantemente monitorato e regolato in base alla produzione di anidride carbonica all'interno del bioreattore, non essendo quest'ultima costante nel tempo. Nella configurazione ex-situ è possibile alimentare idrogeno e anidride carbonica in rapporto stechiometrico, ma porta ad avere rese

in biometano inferiori rispetto all'impianto in-situ. Un limite di entrambe le configurazioni è il basso coefficiente di scambio di materia all'interfaccia gas-liquido [8].

1.3.2 Principali configurazioni reattoristiche

Uno degli aspetti più studiati riguardanti la BHM è la scelta del tipo di reattore nel quale far avvenire la reazione 1.1, in quanto esso può influire notevolmente sull'efficienza del processo. In questa sezione viene mostrata una prospettiva introduttiva delle configurazioni reattoristiche più studiate.

Reattori perfettamente miscelati (CSTR)

I reattori perfettamente miscelati (Continuous Stirred-Tank Reactor, CSTR) sono caratterizzati dalla miscelazione meccanica di un holdup di liquido in cui sono sospesi i microrganismi; la fase gassosa viene invece alimentata al fondo del reattore (Fig. 1.5).

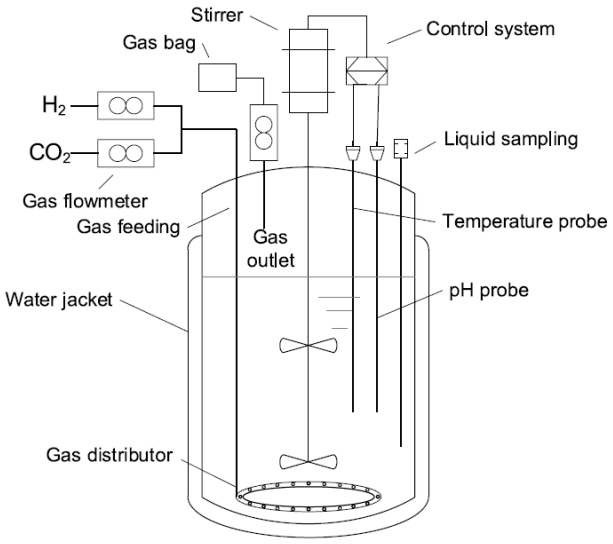


Figura 1.5: Schema di un CSTR per biometanazione [11].

La miscelazione riesce a contrastare il limite della diffusione dell'idrogeno nella fase liquida; tuttavia, per ottenere una miscelazione efficace sono necessari dei rotori che operano a velocità ben più alte di quelle raggiungibili dai reattori di scala industriale [12].

Colonne a bolle (BCR)

Le colonne a bolle (Bubble Column Reactor, BCR) sono ampiamente diffuse nell'ambito dei sistemi multifase. In questo tipo di reattori la fase continua è quella liquida, nella quale sono sospesi i microrganismi [13]; la fase gassosa è invece dispersa in quella liquida sotto forma di bolle ed è alimentata al fondo del reattore (Fig. 1.6).

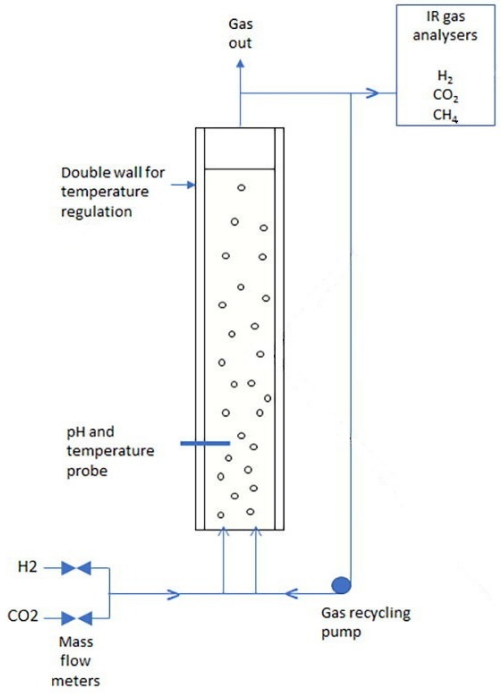


Figura 1.6: Schema di un BCR per biometanazione [13].

Grazie alla dispersione del gas nel liquido, questo reattore è caratterizzato da un'elevata superficie specifica; tuttavia, l'holdup del gas in questa configurazione è basso.

Plug flow reactor (PFR)

Questo tipo di reattore consiste in un tubo di sezione circolare in cui vengono alimentate entrambe le fasi; i microrganismi sono invece depositati sulle pareti del reattore formando un biofilm che le riveste (Fig.1.7) [14].
Con questa configurazione si possono raggiungere delle ottime rese senza incontrare i limiti della diffusione di idrogeno nella fase liquida [14]; tuttavia, dato che la

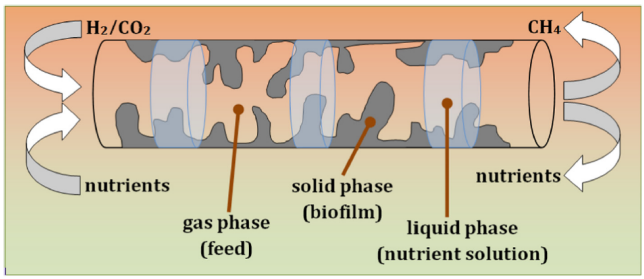


Figura 1.7: Schema di un PFR per biometanazione [14].

reazione avviene solo in prossimità del biofilm che si forma solo sulle pareti del reattore, è molto difficile realizzare uno scale-up di questa soluzione.

Reattori a gocciolamento (TBR)

I reattori a gocciolamento (Trickle-Bed Reactor, TBR) sono dei reattori trifase a letto fisso: al loro interno è presente una struttura di supporto sulla quale è presente il biofilm, la fase gassosa è alimentata dal fondo del reattore mentre quella liquida viene dispersa dall'alto sotto forma di gocce con lo scopo di bagnare il letto impaccato (Fig. 1.8).

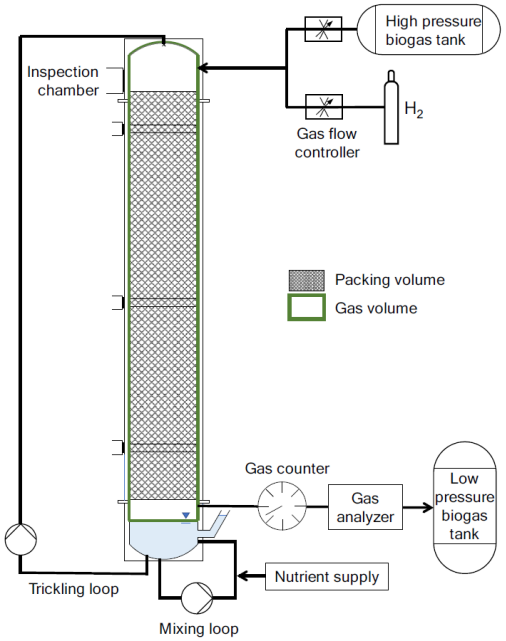


Figura 1.8: Schema di un TBR per biometanazione [15].

A differenza del CSTR, in questo tipo di reattore la fase continua è quella gassosa. Questa configurazione è considerata tra le più efficienti per via dell'elevato holdup di gas e dell'elevata porosità del letto; tuttavia, sono necessari ulteriori studi di fattibilità per poter attuare uno scale-up che si integri bene in un sistema power-to-gas [16].

1.3.3 Simulazioni CFD su TBR per BHM

Nell'ambito dell'ingegneria chimica, i mezzi porosi rivestono un ruolo chiave nell'implementazione di varie tecnologie industriali. La loro porosità e la loro elevata superficie specifica li rendono degli elementi interessanti per una larga serie di applicazioni, tra le quali figurano: l'abbattimento di inquinanti, la catalisi di svariati processi chimici e l'impiego nelle batterie e nelle fuel cells. Questo lavoro di tesi si pone l'obiettivo di approfondire lo studio dell'impiego dei mezzi porosi come catalizzatori; in particolare, verrà studiato il loro comportamento nei sistemi multifase. Come accennato nella sezione precedente, l'impiego dei reattori trickle-bed per lo svolgimento della reazione di biometanazione è stato ritenuto dalla comunità scientifica come una delle opzioni più promettenti per l'implementazione del sistema power-to-gas. Questo tipo di reattori è costituito da un sistema trifase: la fase solida è composta da un letto fisso (mezzo poroso) ed è attraversata da un fluido bifase. Nell'ambito della biometanazione, la fase solida è definita dai microrganismi che fungono anche da catalizzatore della reazione, mentre i reagenti e i prodotti costituiscono il fluido bifase che attraversa il mezzo poroso. Il presente lavoro è stato realizzato dunque con l'intento di approfondire lo studio e la modellazione dei mezzi porosi attraversati da un fluido multifase, essendo essi l'elemento costituente dei reattori trickle-bed. In letteratura sono presenti degli studi fatti sulla modellazione macroscala di un TBR per biometanazione; è stato mostrato che è possibile creare un modello alla macroscala che restituisce risultati concordi con i dati sperimentali [17].
Questo lavoro rappresenta uno studio preliminare sulla modellazione alla microscala di un reattore trickle-bed: l'obiettivo è quello di sfruttare dei modelli alla microscala per ricavare delle proprietà da fornire al modello alla macroscala, riducendo così notevolmente i costi computazionali del modello complessivo. Per svolgere le analisi fluidodinamiche alla microscala, sono state generate e studiate diverse geometrie di conformazione randomica e periodica; esse rappresentano una particella elementare del letto impaccato e sono costituite da delle sfere di raggio costante. Sono state svolte delle simulazioni di fluidodinamica computazionale (computational fluid dynamics, CFD) grazie all'utilizzo del software open-source `OpenFOAM`. Le analisi hanno lo scopo di verifica l'efficacia dei modelli teorici alla microscala nel rappresentare il sistema multifase oggetto di studio. I modelli che cercano di predire il comportamento di un sistema bifase risultano essere molto

complessi: tali modelli comprendono un set di equazioni differenziali alle derivate parziali per ognuna delle due fasi e un ulteriore set di equazioni per calcolare il comportamento dell'interfaccia tra le due fasi, che viene trattata dunque come un'entità a sé stante. Questo implica che le simulazioni che sfruttano questi modelli sono altamente dispendiose in termini di risorse computazionali. Per far fronte a questo limite, è stata valutata l'idea di accoppiare la campagna di simulazioni di CFD alla costruzione di un modello data-driven. Pertanto, nel presente lavoro si è svolto un iniziale tentativo di *fitting* dei dati ricavati dalle simulazioni di CFD realizzato tramite degli interpolatori non lineari. Il fitting permetterebbe di ottenere, con un opportuno grado di tolleranza, dei risultati attendibili riguardanti dei casi che non sono stati direttamente studiati con delle simulazioni di CFD. L'accoppiamento tra i modelli alla microscala e i modelli surrogati potrebbe ridurre significativamente i tempi e le risorse computazionali necessari allo svolgimento di uno studio completo del sistema oggetto di questo lavoro.
Essendo la presente tesi uno studio preliminare sul comportamento dei fluidi bifase in reattori trickle-bed, nel presente lavoro non verranno presi in considerazione gli scambi di calore e di materia che si verificano a causa della reazione di biometanazione. Pertanto, per semplicità, il fluido bifase oggetto delle simulazioni sarà costituito da aria in cui è presente una determinata frazione volumica d'acqua α_{water}.

Capitolo 2

Modelli Teorici

In questo capitolo sono presentati i fondamenti teorici su cui si basa questo lavoro di tesi. Nella sezione 2.1 vengono presentati dei cenni di fluidodinamica utili a descrivere il modello fisico che è stato simulato grazie ai software di CFD. Successivamente, nella sezione 2.2, viene mostrata un'introduzione ai modelli data-driven, struttura portante del machine learning.

2.1 Modello fisico

Il presente lavoro si pone l'obiettivo di studiare il trasporto di quantità di moto che avviene in un flusso bifase quando attraversa un mezzo poroso. Tale fenomeno è ben descritto dalle equazioni di Navier-Stokes, un insieme di equazioni differenziali alle derivate parziali; è possibile facilitarne la risoluzione adottando delle opportune ipotesi semplificative.

2.1.1 Ipotesi adottate

La scelta delle ipotesi semplificative è essenziale: esse permettono di semplificare il sistema senza rinunciare all'accuratezza delle soluzioni ottenute dal sistema semplificato. Sono state adottate le seguenti ipotesi:

- il fluido studiato è un sistema bifase. Tale sistema è costituito da una fase liquida (acqua) ed una fase gassosa (aria);
- le due fasi non scambiano materia tra di loro e il loro contatto non dà luogo a reazioni chimiche;
- entrambe le fasi sono considerate continue;
- la fase liquida assume un comportamento di tipo newtoniano;

– il sistema è considerato isotermo.

2.1.2 Equazioni di governo

Le equazioni che devono essere risolte per descrivere il flusso nel mezzo poroso oggetto di questo lavoro sono:

– equazione di continuità, che descrive la conservazione della massa all'interno di un sistema. Per un fluido incomprimibile, l'equazione presenta la seguente forma:

$$\nabla \cdot \mathbf{U} = \mathbf{0} \tag{2.1}$$

dove $\mathbf{U}$ è il vettore velocità del fluido (m s^{-1}), che si può esprimere tramite le tre coordinate spaziali.

$$\mathbf{U} = \begin{pmatrix} U_x \\ U_y \\ U_z \end{pmatrix} \tag{2.2}$$

∇, invece, è un operatore differenziale esprimibile in coordinate cartesiane:

$$\nabla = \hat{\mathbf{i}} \frac{\partial}{\partial x} + \hat{\mathbf{j}} \frac{\partial}{\partial y} + \hat{\mathbf{k}} \frac{\partial}{\partial z} \tag{2.3}$$

– equazioni di Navier-Stokes, che descrivono il trasporto di quantità di moto (*qdm*) di un sistema. Sono costituite da un primo termine che rappresenta l'accumulo di *qdm* ed il suo trasporto convettivo all'interno del sistema, un secondo termine relativo al trasporto viscoso di *qdm* e da due termini rappresentanti il contributo di *qdm* generato da forze esterne. Per un fluido bifase costituito da due fasi immiscibili e newtoniane [18]:

$$\rho \frac{D\mathbf{U}}{Dt} - \nabla \cdot (\mu \nabla \mathbf{U}) - (\nabla \mathbf{U}) \cdot \nabla \alpha = -\nabla p - \mathbf{g} \nabla \rho + \sigma \kappa \nabla \alpha \tag{2.4}$$

dove ρ è la densità del fluido (kg m^{-3}), μ è la viscosità dinamica del fluido (Pa s), α è la frazione d'acqua (-), p è la pressione (Pa), σ è la tensione superificiale (N m^{-1}) e κ rappresenta la curvatura dell'interfaccia (m^{-1}); al primo termine è presente la *derivata sostanziale* della velocità, espressa come segue:

$$\frac{D}{Dt}\mathbf{U} = \frac{\partial}{\partial t}\mathbf{U} + (\mathbf{U} \cdot \nabla)\mathbf{U} \tag{2.5}$$

Nel secondo termine, invece, è presente l'operatore differenziale ∇^2, chiamato *laplaciano* e definito come:

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2} \tag{2.6}$$

La curvatura media dell'interfaccia κ può essere approssimata come segue:

$$\kappa = -\nabla \cdot \left(\frac{\nabla \alpha}{|\nabla \alpha|} \right) \tag{2.7}$$

– equazione di bilancio all'interfaccia, che garantisce la conservazione della frazione d'acqua α all'interno del sistema. L'equazione si presenta nella seguente forma, espressa esplicitando la fase liquida:

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot (\mathbf{U}_l \alpha) = 0 \tag{2.8}$$

dove $\mathbf{U}_l$ è il vettore della velocità della fase liquida (m s^{-1}).

Gli algoritmi sfruttati dal software di fluidodinamica computazionale che è stato impiegato per questo lavoro adottano il modello *volume of fluid* (VOF): secondo tale modello, le proprietà del flusso bifase possono essere approssimate alla media pesata delle proprietà dei singoli flussi delle due fasi, utilizzando la frazione di una delle due fasi α come peso. Pertanto, le proprietà presentate nelle equazioni di governo sono descritte, nell'ambito del modello VOF, nel seguente modo:

$$\mathbf{U} = \alpha \mathbf{U}_l + (1 - \alpha)\mathbf{U}_g \tag{2.9}$$

$$\rho = \alpha \rho_l + (1 - \alpha)\rho_g \tag{2.10}$$

$$\mu = \alpha \mu_l + (1 - \alpha)\mu_g \tag{2.11}$$

In questa trattazione, con α si definisce la frazione della fase liquida, rappresentata dall'acqua.
L'utilizzo delle reti neurali richiede di identificare una grandezza che sia rappresentativa dei risultati delle simulazioni. In questo lavoro viene studiato il flusso di un fluido multifase attraverso un mezzo poroso, pertanto il parametro scelto per rappresentare i risultati delle simulazioni è la *permeabilità* del fluido nel mezzo poroso.

Permeabilità La permeabilità (κ) è definita come la capacità di un mezzo poroso di essere attraversato da un fluido. Può essere valutata tramite la legge di Darcy, che descrive il moto di un fluido attraverso un mezzo poroso:

$$u_s = \frac{\kappa}{\mu} \frac{\Delta P}{L} \tag{2.12}$$

dove u_s è la velocità superficiale del fluido (m s^{-1}) , μ è la viscosità dinamica del fluido (Pa s), ΔP è il gradiente di pressione (Pa) e L è la lunghezza della

dimensione del sistema parallela a u_s; dalla formulazione della legge di Darcy si può notare che l'unità di misura della permeabilità κ sia (m^2).
Se nel sistema sono presenti due o più fluidi, risulta opportuno poter descrivere la capacità del mezzo poroso di far scorrere preferibilmente uno dei fluidi presenti rispetto agli altri; si parlerà dunque di permeabilità relative. Il sistema studiato in questo lavoro è caratterizzato dalla presenza di due fluidi, aria e acqua; la permeabilità di un fluido rispetto all'altro è funzione delle saturazioni dei due fluidi [19]. Le permeabilità relative sono quindi rappresentate da una coppia di curve dipendenti dalle saturazioni dei due fluidi: all'aumentare della saturazione di uno dei due fluidi, la sua permeabilità relativa aumenta. La legge di Darcy, nel caso di un sistema multifase, assumerà la seguente forma:

$$u_{s,i} = K \frac{\kappa_i}{\mu_i} \frac{\partial p_i}{\partial x} \tag{2.13}$$

dove K è la permeabilità assoluta del mezzo poroso, l'indice i rappresenta la i-esima fase presente nel sistema e k_i è la permeabilità relativa della fase i-esima.

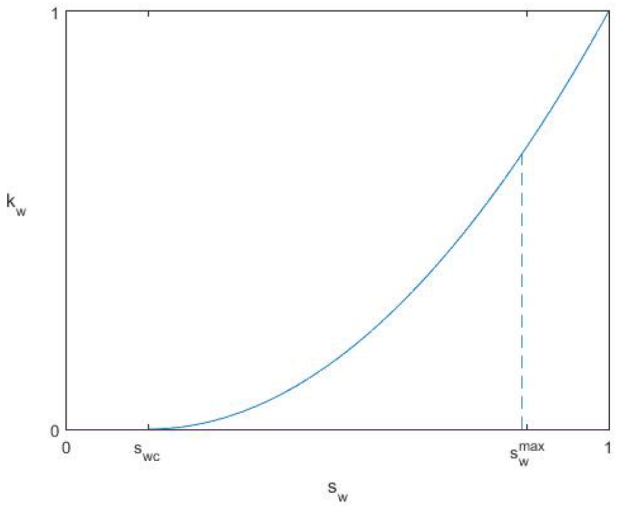


Figura 2.1: Esempio di una curva di permeabilità relativa [19].

In Fig.(2.1) è mostrato l'andamento della permeabilità relativa di una fase (acqua, in un sistema acqua-petrolio [19]) in funzione della sua saturazione. Si può notare come la permeabilità relativa dell'acqua κ_w tenda a quella che si avrebbe in un sistema monofase quando la saturazione d'acqua s_w tende al valore unitario.

2.2 Modelli data-driven

Esistono dei modelli, detti *data-driven*, che permettono di esprimere delle relazioni funzionali tra le variabili in ingresso e quelle in uscita di un dato sistema senza ricavare delle equazioni esplicite che ne descrivono la fisica. Per costruire un modello data-driven è necessario disporre di un dataset contenente tutte le informazioni riguardanti i parametri di input e di output desiderati. La creazione di un modello data-driven è caratterizzata dalle seguenti fasi:

- ottenimento dei dati: ogni dato, definito *sample*, possiede un certo numero di proprietà di input (*feature*) ed eventualmente degli output (*label* o *target*);
- *pre-processing* dei dati: in questa fase viene analizzato ogni sample e si verifica la presenza di *outlier* o l'assenza di alcune feature. Inoltre, durante questa fase avviene lo *scaling* dei dati;
- allenamento del modello (*training*): dopo aver selezionato delle adeguate feature da utilizzare in questa fase, il modello viene allenato ricevendo come input alcuni sample. Il training serve a ottimizzare i parametri del modello, migliorandone la capacità predittiva;
- interrogazione del modello (*testing*): in questa fase vengono fornite delle feature appartenenti a dei sample che non sono stati utilizzati in fase di training e il modello restituisce una predizione dei corrispettivi target. Il testing viene adoperato per valutare le capacità predittive del modello tramite alcuni indicatori statistici.

È possibile suddividere i modelli in base al tipo di supervisione a cui sono sottoposti durante la fase di training; si fa distinzione, quindi, tra:

- *supervised learning*: il training di questi modelli avviene tramite dati che sono provvisti sia di feature che di label. Questo tipo di training è utilizzato per la creazione di modelli di classificazione (distinzione di un elemento tra un numero finito di categorie) o di regressione (predizione di un valore numerico);
- *unsupervised learning*: modelli allenati con dei dati che non possiedono label. A questa categoria appartengono i modelli di clustering (suddivisione del dataset in sottogruppi contraddistinti da delle caratteristiche comuni), di determinazione di dati anomali o di visualizzazione (identificazione di pattern all'interno del dataset);
- *semisupervised learning*: un ibrido tra i due modelli descritti nei punti precedenti. Pertanto, questi modelli sono allenati con un dataset in cui solo una parte dei dati possiede label;

– *reinforcement learning*: modelli basati su un meccanismo di benefici e perdite. Durante il training, il modello riceve dei feedback in base alla qualità della predizione che modificano il comportamento del modello stesso.

Nell'ambito di questo lavoro di tesi è stato utilizzato il supervised learning per la creazione di un modello predittivo di regressione. Tale modello ha il compito di predire il valore di una variabile target (la permeabilità κ) sulla base dei valori delle feature ($\Delta P/\Delta L$ e α_{water}) ottenuti grazie alle simulazioni di fluidodinamica computazionale, data una determinata geometria (mezzo poroso).

2.2.1 Normalizzazione delle feature

Uno degli step più importanti del pre-processing dei dati è la normalizzazione (*scaling*) delle feature. Questa operazione permette a tutte le feature di contribuire in modo significativo alla predizione del modello. Inoltre, è stato verificato che la normalizzazione delle feature velocizza la convergenza di alcuni algoritmi di machine learning, come il gradient descent [20]. Di seguito vengono riportate alcune delle tecniche di scaling utilizzate nell'ambito del machine learning; viene definito x il vettore dei valori di una determinata feature per ogni sample, mentre con x' viene indicato il vettore normalizzato:

– normalizzazione min-max: tutti i valori del vettore x vengono scalati affinché assumano dei valori compresi tra un valore minimo a e uno massimo b

$$x' = a + \frac{(x - \min(x))\,(b - a)}{\max(x) - \min(x)} \tag{2.14}$$

dove $\min(x)$ e $\max(x)$ rappresentano, rispettivamente, il valore minimo e il valore massimo del vettore x;

– normalizzazione media: ai valori viene sottratto il valore medio $\bar{x}$ e, successivamente, ogni valore viene diviso per il range del vettore x

$$x' = \frac{x - \bar{x}}{\max(x) - \min(x)} \tag{2.15}$$

– standardizzazione: simile al metodo precedente, ma la normalizzazione avviene dividendo per la deviazione standard

$$x' = \frac{x - \bar{x}}{\sigma} \tag{2.16}$$

dove σ rappresenta la deviazione standard, definita come:

$$\sigma = \frac{\sqrt{\sum_{i=1}^{N}(x_i - \bar{x})^2}}{N} \tag{2.17}$$

– normalizzazione all'unità di lunghezza: tutti i valori del vettore x vengono scalati affinché la lunghezza del vettore normalizzato x' sia pari a uno

$$x' = \frac{x}{\|x\|_2} = \frac{x}{\sqrt{x_1^2 + x_2^2 + \cdots + x_N^2}} \tag{2.18}$$

2.2.2 Modelli di regressione

I modelli di regressione sono dei supervised model che hanno l'obiettivo di predire il valore di una variabile target sulla base dei valori delle feature fornite come variabile di input. Tali modelli si suddividono in due categorie, in base al numero di regressori (feature) utilizzati:

– regressione semplice: il valore della variabile target è predetto sulla base del valore di un singolo regressore.

$$y = f(x_1) \tag{2.19}$$

– regressione multipla: vengono utilizzate n feature per la predizione del valore target.

$$y = f(x_1, x_2, \cdots, x_n) \tag{2.20}$$

dove x_i rappresenta la generica feature e y corrisponde al valore della variabile target.

Inoltre, si può distinguere tra regressione lineare e non lineare, sulla base delle relazioni che sono presenti tra la variabile target e le varie feature.
L'esempio più semplice di modello di regressione è il modello lineare semplice, descritto dalla seguente equazione:

$$\hat{y} = \beta_0 + \beta_1 x_1 \tag{2.21}$$

dove $\hat{y}$ corrisponde al valore predetto della variabile target e β_0 ,β_1 sono i coefficienti di regressione. L'obiettivo della fase di training del modello è quella di ottimizzare i coefficienti di regressione, determinando il loro valore per il quale si minimizza l'errore tra il valore target reale e quello predetto dal modello stesso. Per minimizzare tale errore, è necessario calcolare il valore dei vari *residui* del modello, definiti nel seguente modo:

$$r_i = y_i - \hat{y}_i \tag{2.22}$$

dove y_i è il valore reale della variabile target, mentre $\hat{y}_i$ rappresenta il valore della variabile target predetto dal modello di regressione. Minimizzando la somma dei

quadrati dei residui del modello si ottimizzano i coefficienti di regressione; questo metodo è chiamato *Least Squares Method* (LSM):

$$\min\left(\sum_i (r_i)^2\right) = \min\left(\sum_i (y_i - (\beta_0 + \beta_1 x_1))^2\right) \tag{2.23}$$

Estendendo il concetto ad un numero n di regressori, si ottiene un modello lineare multiplo:

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n \tag{2.24}$$

Il singolo coefficiente β_i rappresenta l'effetto medio su $\hat{y}$ della variazione unitaria del regressore x_i, mantenendo il resto costante. Per questo motivo, i singoli coefficienti di regressione vanno stimati e testati separatamente; pertanto, è cruciale che le feature scelte per creare il modello siano indipendenti tra loro e non correlati.
Esistono vari metodi per costruire dei modelli di regressione; nell'ambito di questo lavoro di tesi si porrà l'attenzione sull'utilizzo delle reti neurali (*artificial neural networks*, ANN). Nel presente lavoro è stato costruito un modello di regressione basato sulla struttura MLP (MultiLayer Perceptron) utilizzando gli strumenti forniti dalle librerie `Python scikit-learn` [21] e `Keras` [22]. La valutazione dell'accuratezza delle predizioni del modello è basata sulle seguenti metriche:

– Mean Absolute Error (MAE): definisce l'errore medio assoluto

$$MAE = \frac{1}{n}\sum_i |r_i| = \frac{1}{n}\sum_i |y_i - \hat{y}_i| \tag{2.25}$$

– Mean Squared Error (MSE): definisce l'errore quadratico medio

$$MSE = \frac{1}{n}\sum_i (r_i)^2 = \frac{1}{n}\sum_i (y_i - \hat{y}_i)^2 \tag{2.26}$$

– R^2: coefficiente che rappresenta la qualità del fitting dei dati

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y_i})^2} \tag{2.27}$$

dove $\bar{y} = \frac{1}{n}\sum_i y_i$

2.2.3 Reti neurali

Le reti neurali costituiscono una categoria di algoritmi di machine learning indicati per sistemi multidimensionali e non lineari. L'unità base delle reti neurali è detta *Threshold Logic Unit* (TLU, Fig.2.2). Questa unità è connessa ai valori di input

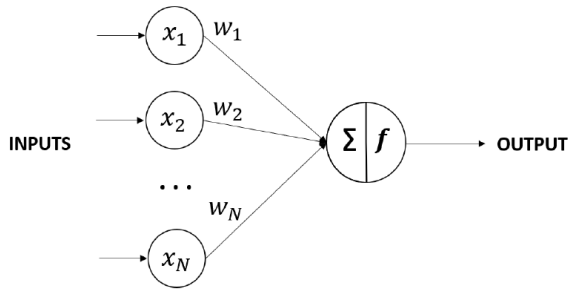


Figura 2.2: Schema di una threshold logic unit [23].

tramite dei pesi; la TLU esegue la somma pesata degli input e ne applica una funzione. Il processo appena descritto può essere rappresentato con la seguente equazione:

$$z = f(w_1 x_1 + w_2 x_2 + \cdots + w_N x_N) = f(\mathbf{w}^T \mathbf{x}) \tag{2.28}$$

dove $(x_1, x_2, \cdots, x_N)$ sono gli input e $\mathbf{x}$ il loro vettore, mentre $(w_1, w_2, \cdots, w_N)$ e $\mathbf{w}$ rappresentano i pesi e il loro vettore. La funzione f invece è detta *funzione di attivazione* e vengono mostrate di seguito le più utilizzate:

– funzione *Rectified Linear Unit* (ReLU)

$$ReLU(z) = \max(0, z) \tag{2.29}$$

– funzione logistica

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{2.30}$$

– funzione tangente iperbolica

$$\tanh(z) = \frac{\sinh(z)}{\cosh(z)} = \frac{1 - e^{-2z}}{1 + e^{-2z}} \tag{2.31}$$

La struttura MLP è caratterizzata dalla presenza di una rete di unità dette *percettroni*; ogni percettrone elabora una serie di informazioni ricevute in ingresso e restituisce un valore di output che sarà selezionato come input dell'unità successiva. Un percettrone è costituito da un *layer* di TLU, dove ogni TLU è connessa a ciascun input (Fig. 2.4) Esprimendo i calcoli eseguiti dalle varie TLU, è possibile descrivere un percettrone nel seguente modo:

$$\begin{aligned} z_1 &= f(w_{1,1}\, x_1 + w_{1,2}\, x_2 + \cdots + w_{1,N}\, x_N + b_1) = f(\mathbf{w}_1^T\, \mathbf{x}) \\ z_2 &= f(w_{2,1}\, x_1 + w_{2,2}\, x_2 + \cdots + w_{2,N}\, x_N + b_2) = f(\mathbf{w}_2^T\, \mathbf{x}) \\ &\vdots \\ z_N &= f(w_{N,1}\, x_1 + w_{N,2}\, x_2 + \cdots + w_{N,N}\, x_N + b_N) = f(\mathbf{w}_N^T\, \mathbf{x}) \end{aligned} \tag{2.32}$$

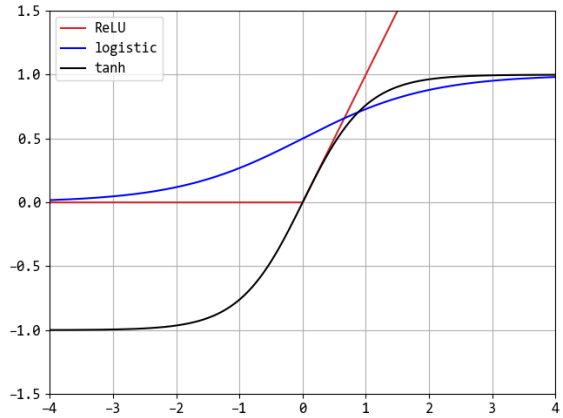


Figura 2.3: Principali funzioni di attivazione.

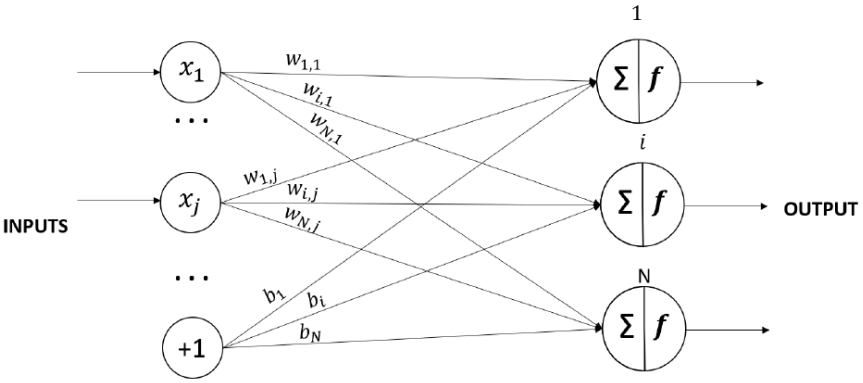


Figura 2.4: Schema di un percettrone [23].

In forma matriciale:

$$\mathbf{z} = f(\mathbf{W}\mathbf{x} + \mathbf{b}) \tag{2.33}$$

dove $\mathbf{z}$ è il vettore degli output, $\mathbf{W}$ è la matrice dei pesi, $\mathbf{x}$ è il vettore degli input e $\mathbf{b}$ è il vettore dei *bias*; il bias è una feature aggiuntiva ($x_0 = 1$) che viene aggiunta in ogni layer e che non riceve alcun valore in ingresso. Essendo indipendente dalle feature scelte per la costruzione del modello, il bias assume la stessa funzione dell'intercetta in una relazione lineare.
Aggiungendo nuovi layer alla struttura precedentemente descritta, si definisce un'architettura denominata MLP (multilayer perceptron); esse è costituita da un layer di input nel quale sono forniti i valori delle feature, una serie di *hidden layer* costituiti da dei percettroni di dimensione variabile e un layer di output che restituisce il valore del target predetto. Se un'ANN presenta due o più hidden layer,

essa viene definita *Deep Neural Network* (DNN). In Fig.(2.5) è mostrato un esempio di rete MLP costituita dai layer di input e output e da due hidden layer. I calcoli eseguiti da questa rete possono essere descritti nel seguente modo:

$$\mathbf{z}^{III} = f^{III}\{\mathbf{W}^{III} f^{II}[\mathbf{W}^{II} f^{I}(\mathbf{W}^{I}\mathbf{x} + \mathbf{b}^{I}) + \mathbf{b}^{II}] + \mathbf{b}^{III}\} \tag{2.34}$$

dove gli apici indicano il numero del layer corrispondente. Il training delle DNN viene svolto attraverso un algoritmo di *retropropagazione*. Per ogni sample del dataset, viene calcolata una predizione che viene confrontata con il valore atteso del label; dopo aver calcolato l'errore della predizione, l'algoritmo di retropropagazione ripercorre la rete a ritroso (dal layer di output verso quello di input) e aggiorna iterativamente i valori dei pesi e dei bias con lo scopo di minimizzare l'errore.

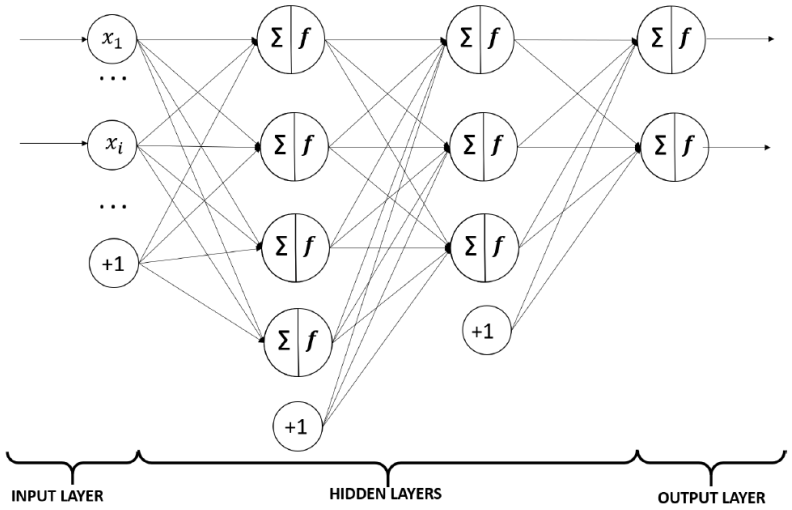


Figura 2.5: Schema di un esempio di rete MLP in cui sono presenti due hidden layer [23].

Capitolo 3

Metodi Computazionali

In questo capitolo sono descritti i fondamenti della fluidodinamica computazionale. Nella sezione 3.1 viene introdotto il metodo dei volumi finiti (FVM) e viene descritto l'approccio utilizzato per studiare campi scalari e vettoriali risolvendo le equazioni di Navier-Stokes. Nelle sezioni successive vengono mostrati i metodi di discretizzazione spaziale e temporale del dominio computazionale. Infine, nella sezione 3.5, vengono presentati alcuni cenni relativi al software open-source `OpenFOAM`, utilizzato in questo lavoro per implementare il FVM. Le nozioni di CFD presentate in questo capitolo sono tratte dal libro *"The Finite Volume Method in Computational Fluid Dynamics"* scritto da F.Moukalled, L.Mangani, M.Darwish [24].

3.1 Metodo ai volumi finiti

Il metodo dei volumi finiti (FVM) è stato introdotto nella fluidodinamica computazionale dopo altri approcci numerici, come il metodo delle differenze finite e quello degli elementi finiti. La sua diffusione è dovuta a diversi vantaggi significativi, primo fra tutti la capacità di conservare in modo intrinseco proprietà fondamentali come massa ed energia. Un ulteriore punto di forza del FVM è la sua grande flessibilità: infatti, la discretizzazione può essere effettuata direttamente nello spazio fisico, senza la necessità di trasformare il dominio fisico in un dominio computazionale [25]. Il processo di discretizzazione avviene simultaneamente a livello geometrico e a livello delle equazioni di trasporto, come schematizzato in Fig.(3.1). La suddivisione dell'intera geometria in celle, chiamata *meshing* del dominio computazionale, è un passaggio fondamentale. Parallelamente, anche le equazioni di trasporto devono essere discretizzate e risolte per ogni cella. Poiché le celle sono interconnesse, non possono essere trattate come entità indipendenti, ma

devono scambiarsi informazioni con i volumi adiacenti, garantendo una corretta trasmissione di dati attraverso il dominio computazionale.

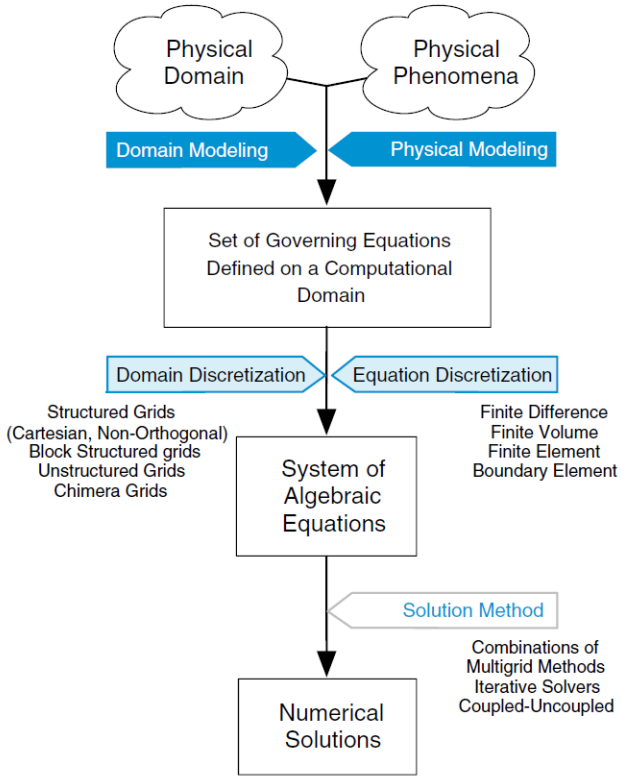


Figura 3.1: Schema del processo di discretizzazione [24].

3.1.1 Equazioni di governo

Il primo passo del FVM consiste nella formulazione delle equazioni semi-discretizzate, partendo dalle equazioni di trasporto che descrivono il sistema da simulare. In questa sezione, per semplicità, le equazioni presentate vengono espresse in termini di trasporto di un generico scalare:

$$\frac{\partial(\rho\phi)}{\partial t} + \nabla \cdot (\rho \mathbf{U}\phi) = \nabla \cdot (\Gamma^{\phi} \nabla\phi) + Q^{\phi} \tag{3.1}$$

dove ϕ rappresenta il generico scalare, Γ^{ϕ} ($\mathrm{m}^2\ \mathrm{s}^{-1}$) il coefficiente di diffusività generico e Q^{ϕ} il termine sorgente dello scalare. Se il sistema è stazionario, l'equazione assume la seguente forma:

$$\nabla \cdot (\rho \mathbf{U}\phi) = \nabla \cdot (\Gamma^{\phi} \nabla\phi) + Q^{\phi} \tag{3.2}$$

Se consideriamo una cella C ed integriamo l'Eq.(3.1) sul volume della cella V_C, applicando il teorema di Gauss al termine convettivo e a quello diffusivo si ottiene la seguente equazione:

$$\oint_{\partial V_C} (\rho \mathbf{U}\phi) \cdot d\mathbf{S} = \oint_{\partial V_C} (\Gamma^\phi \nabla\phi) \cdot d\mathbf{S} + \int_{V_C} Q^\phi dV \tag{3.3}$$

dove $\mathbf{S}$ è il vettore superficie e $\oint_{\partial V_C}$ rappresenta l'integrale di superficie sul volume V_C.
Introducendo il concetto di flusso convettivo e diffusivo, rispettivamente $\mathbf{J}^{\phi,C}$ e $\mathbf{J}^{\phi,D}$:

$$\mathbf{J}^{\phi,C} = \rho \mathbf{U}\phi \tag{3.4}$$

$$\mathbf{J}^{\phi,D} = -\Gamma^\phi \nabla\phi \tag{3.5}$$

$$\mathbf{J}^{\phi} = \mathbf{J}^{\phi,C} + \mathbf{J}^{\phi,D} \tag{3.6}$$

è possibile discretizzare l'integrale superficiale sulla cella C sostituendolo con la somma dei termini di flusso relativi alle facce di C. Si ottengono dunque, le seguenti espressioni per gli integrali superficiali del flusso convettivo, diffusivo e totale:

$$\oint_{\partial V_C} \mathbf{J}^{\phi,C} \cdot d\mathbf{S} = \sum_{f \sim faces(V_C)} \left(\int_f (\rho \mathbf{U}\phi) \cdot d\mathbf{S}) \right) \tag{3.7}$$

$$\oint_{\partial V_C} \mathbf{J}^{\phi,D} \cdot d\mathbf{S} = \sum_{f \sim faces(V_C)} \left(\int_f (\Gamma^\phi \nabla\phi) \cdot d\mathbf{S}) \right) \tag{3.8}$$

$$\oint_{\partial V_C} \mathbf{J}^{\phi} \cdot d\mathbf{S} = \sum_{f \sim faces(V_C)} \left(\int_f \mathbf{J}_f^{\phi} \cdot d\mathbf{S}) \right) \tag{3.9}$$

Adesso è possibile ottenere la forma semi-discretizzata dell'Eq.(3.3):

$$\sum_{f \sim nb(C)} (\rho \mathbf{U}\phi - \Gamma^\phi \nabla\phi)_f \cdot \mathbf{S}_f = Q_C^\phi V_C \tag{3.10}$$

3.1.2 Costruzione del sistema

Lo step successivo consiste nel trasformare l'Eq.(3.10) in un sistema di equazioni esprimendo i flussi in funzione dei valori delle variabili memorizzate nei centroidi delle celle adiacenti alla cella C.
Facendo riferimento alla Fig.(3.2), prendiamo in considerazione in prima analisi l'interazione tra la cella C e la cella adiacente F_1. Il flusso f_1 tra le due celle può essere scomposto nel seguente modo:

$$\mathbf{J}_f^{\phi} \cdot \mathbf{S}_f = FluxT_f = FluxC_f \phi_C + FluxF_f \phi_F + FluxV_f \tag{3.11}$$

dove il flusso totale $FluxT_f$ è stato decomposto in tre termini:

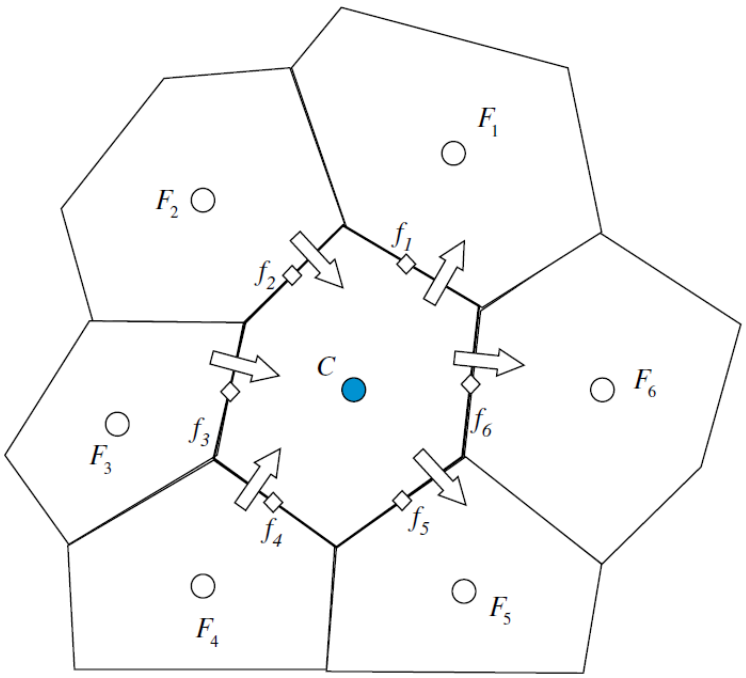


Figura 3.2: Flusso attraverso le facce f della cella C [24].

- $FluxC_f\phi_C$, prodotto tra il coefficiente di linearizzazione di C ed il valore dello scalare ϕ nel centroide della cella stessa;

- $FluxF_f\phi_F$, prodotto tra il coefficiente di linearizzazione di F ed il valore dello scalare ϕ nel centroide della cella stessa;

- $FluxV_f$, termine non lineare del flusso.

I valori dei coefficienti $FluxC_f$, $FluxF_f$ e $FluxV_f$ dipendono dal termine discretizzato e dallo schema di discretizzazione impiegato [25].
Sostituendo l'Eq.(3.11) nell'Eq.(3.10) si può estendere il concetto a tutte le facce adiacenti a C:

$$\sum_{f\sim nb(C)} (\mathbf{J}_f^{\phi} \cdot \mathbf{S}_f) = \sum_{f\sim nb(C)} (FluxC_f\phi_C + FluxF_f\phi_F + FluxV_f) = Q_C^{\phi}V_C \qquad (3.12)$$

Considerando che anche il flusso volumico può essere scomposto in un termine lineare ed uno non lineare:

$$Q_C^{\phi}V_C = FluxT = FluxC\phi_C + FluxV \qquad (3.13)$$

sostituendo l'Eq.(3.13) nell'Eq.(3.12) si ottiene l'equazione algebrica:

$$a_C\phi_C + \sum_{f\sim nb(C)} (a_F\phi_F) = b_C \tag{3.14}$$

dove compaiono tre coefficienti legati ai valori dei coefficienti di linearizzazione del flusso:

$$a_C = \sum_{f\sim nb(C)} FluxC_f - FluxC \tag{3.15}$$

$$a_F = FluxF_f \tag{3.16}$$

$$b_C = \sum_{f\sim nb(C)} FluxV_f - FluxV \tag{3.17}$$

Definendo un'equazione algebrica come Eq.(3.14) per ogni cella del dominio computazionale, si ottiene un sistema di equazioni algebriche che permette di calcolare il campo dello scalare ϕ nell'intero dominio.

3.1.3 Algoritmo di risoluzione

L'ultimo step del processo di discretizzazione prevede la risoluzione del sistema lineare costruito nella sezione precedente, che si presenta nella seguente forma:

$$\mathbf{A}\boldsymbol{\phi} = \mathbf{b} \tag{3.18}$$

dove $\mathbf{A}$ è la matrice dei coefficienti prodotti dal processo di linearizzazione del flusso, che dipendono dallo schema di discretizzazione adottato. $\boldsymbol{\phi}$ contiene il valore del generico scalare ϕ in tutto il dominio computazionale, dunque costituisce il vettore delle incognite. $\mathbf{b}$, infine, è il vettore dei termini noti che comprende i termini sorgente, le condizioni di bordo e i termini non lineari del flusso.
Per risolvere un sistema lineare, espresso qui esplicitando i componenti delle matrici:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N-1} & a_{1N} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{N,1} & a_{N,2} & \cdots & a_{N,N-1} & a_{N,N} \end{bmatrix} \begin{bmatrix} \phi_1 \\ \vdots \\ \phi_N \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ b_N \end{bmatrix} \tag{3.19}$$

esistono diversi metodi che si possono racchiudere in due categorie: i metodi diretti e i metodi iterativi. Gli ultimi sono preferiti nell'ambito della CFD, poiché richiedono meno risorse in termini di memoria e di potenza di calcolo [24].
Uno dei metodi iterativi più utilizzati per risolvere l'Eq.(3.18) è quello di Gauss-Seidel, implementato secondo la seguente formula iterativa:

$$\phi_i^{(n)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij}\phi_j^{(n)} - \sum_{j=i+1}^{N} a_{ij}\phi_j^{(n-1)} \right) \qquad i = 1,2,3,...,N \tag{3.20}$$

3.1.4 Condizioni di bordo

In corrispondenza dei bordi del dominio computazionale è necessario specificare delle condizioni di bordo per poter chiudere il problema. Le due condizioni maggiormente impiegate nel FVM sono:

– Condizione di Dirichlet: si specifica il valore dello scalare ϕ al bordo del dominio.

$$\phi_b = \phi_{b,noto} \tag{3.21}$$

Il flusso al bordo viene calcolato di conseguenza nel seguente modo:

$$\mathbf{J}_b^{\phi} \cdot \mathbf{S}_b = \mathbf{J}_b^{\phi,C} \cdot \mathbf{S}_b = (\rho \mathbf{U} \phi)_b \cdot \mathbf{S}_b = (\rho_b \mathbf{U}_b \cdot \mathbf{S}_b)\phi_b = \dot{m}_f \phi_{b,noto} \tag{3.22}$$

– Condizione di Neumann: viene specificato il valore del flusso dello scalare al bordo del dominio.

$$\mathbf{J}_b^{\phi} \cdot \mathbf{S}_b = \mathbf{J}_b^{\phi} \cdot \mathbf{n}_b S_b = q_{b,noto} S_b \tag{3.23}$$

dove $q_{b,noto}$ è espresso come flusso di ϕ per unità di area.

3.2 Meshing

Il FVM richiede, oltre alla discretizzazione delle equazioni di governo, la definizione di una *mesh* computazionale. Essa consiste in un insieme di celle e rappresenta la discretizzazione di un dominio continuo in una serie finita di elementi; il processo di *meshing* suddivide il dominio in un numero finito di celle e determina le relazioni topologiche tra le celle stesse, che possono comunicare con altre celle adiacenti o con il bordo del dominio computazionale. Le equazioni discretizzate sono dunque definite e risolte per ogni elemento della mesh. Le informazioni riguardanti i campi scalari e vettoriali che sono stati calcolati in una cella vengono memorizzate nel centroide della cella stessa, ovvero nel suo punto centrale. Il centroide, così come il volume della cella e la superficie di ogni sua faccia, viene calcolato e memorizzato in modo da essere successivamente utilizzato durante il calcolo dei flussi volumici e superficiali. Le celle della mesh solitamente sono dei poliedri o dei poligoni (Fig.3.4). La griglia è solitamente classificata come strutturata o non strutturata in base alla possibilità di numerare sequenzialmente, secondo una logica, tutte le celle. I *tools* del software `OpenFOAM` utilizzati in questo lavoro sono in grado di costruire griglie non strutturate di celle poliedriche.

3.3 Discretizzazione spaziale

Per poter risolvere il sistema di equazioni algebriche costruito nelle sezioni precedenti (Eq.3.20) è necessario specificare quale approccio viene utilizzato per discretizzare il laplaciano ed il gradiente nello spazio. Da tale discretizzazione dipendono

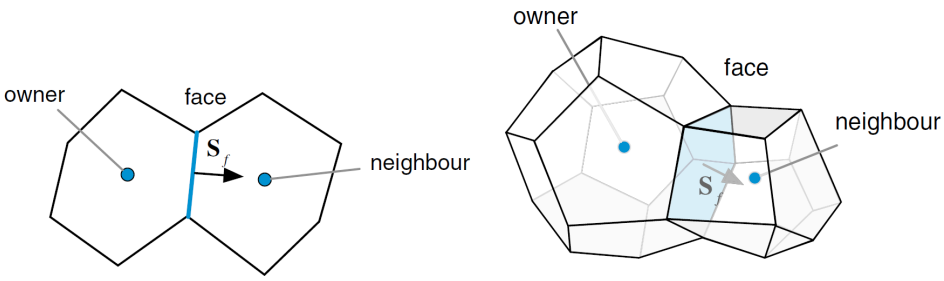


Figura 3.3: Topologia delle celle [24].

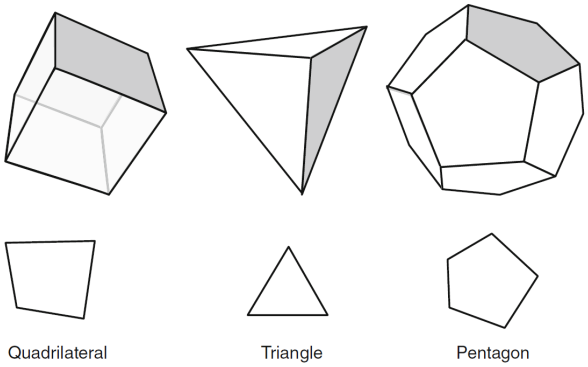


Figura 3.4: Esempi di elementi presenti in mesh tridimensionali o bidimensionali [24].

anche i valori dei coefficienti definiti nell'Eq.(3.14). Pertanto, in questa sezione verrà fornita un'introduzione sulla discretizzazione spaziale. Per semplicità, verrà considerata la griglia bidimensionale cartesiana, mostrata in Fig.(3.5).

3.3.1 Termine diffusivo

Il termine diffusivo delle equazioni di trasporto contiene il laplaciano del generico scalare ϕ e, come mostrato nella sezione 3.1.1, si può esprimere come:

$$\sum_{f \sim nb(C)} (-\Gamma^{\phi} \nabla \phi)_f \cdot \mathbf{S}_f \tag{3.24}$$

Facendo riferimento alla griglia mostrata in Fig.(3.5), è possibile esplicitare la somma nel seguente modo:

$$(-\Gamma^{\phi} \nabla \phi)_e \cdot \mathbf{S}_e + (-\Gamma^{\phi} \nabla \phi)_n \cdot \mathbf{S}_n + (-\Gamma^{\phi} \nabla \phi)_w \cdot \mathbf{S}_w + (-\Gamma^{\phi} \nabla \phi)_s \cdot \mathbf{S}_s \tag{3.25}$$

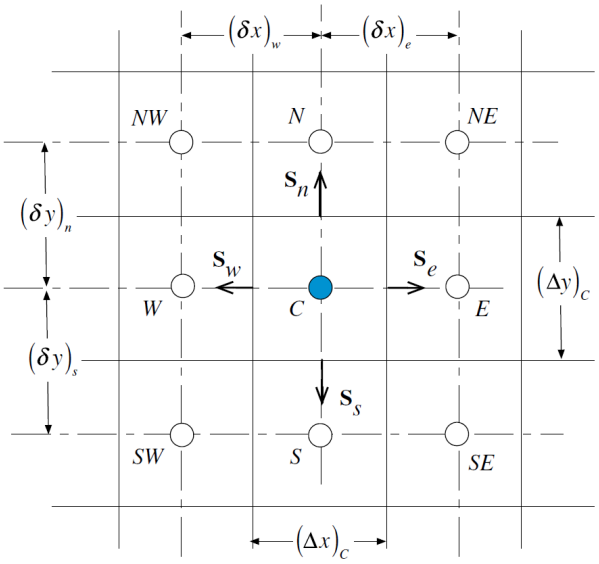


Figura 3.5: Griglia cartesiana [24].

I singoli flussi possono essere calcolati utilizzando la definizione del gradiente; per esempio, il flusso diffusivo nella faccia est si esprime come:

$$\mathbf{J}_e^{\phi,D} = -(\Gamma^\phi \nabla\phi)_e \cdot \mathbf{S}_e = -\Gamma_e^\phi S_e \left(\frac{\partial\phi}{\partial x}\mathbf{i} + \frac{\partial\phi}{\partial y}\mathbf{j}\right)_e \cdot \mathbf{i} = -\Gamma_e^\phi (\Delta y)_e \left(\frac{\partial\phi}{\partial x}\right)_e \tag{3.26}$$

Per poter calcolare il flusso appena descritto è necessario imporre un profilo di ϕ tra i valori dei centroidi delle celle adiacenti, in modo tale da definire un gradiente. Il profilo utilizzato nel termine diffusivo è lineare:

$$\left(\frac{\partial\phi}{\partial x}\right)_e = \frac{\phi_E - \phi_C}{(\delta x)_e} \tag{3.27}$$

Infine, sostituendo l'Eq.(3.27) nell'Eq.(3.26), si ottiene la completa discretizzazione del termine diffusivo:

$$FluxT_e = -\Gamma_e^\phi (\Delta y)_e \frac{\phi_E - \phi_C}{\delta x_e} = \Gamma_e^\phi \frac{(\Delta y)_e}{\delta x_e}(\phi_C - \phi_E) \tag{3.28}$$

Si può notare che l'espressione appena ottenuta è coerente con la definizione di flusso totale descritta nella sezione 3.1.2 (Eq.3.11), essendo la somma di un termine proporzionale a ϕ_C, un termine proporzionale a ϕ_E e un contributo non lineare, in questo caso nullo.

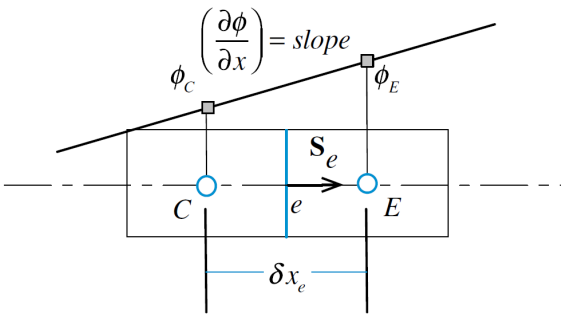


Figura 3.6: Profilo di interpolazione lineare [24].

3.3.2 Termine convettivo

La discretizzazione del termine convettivo risulta più complessa rispetto a quella del termine diffusivo; questo è dovuto alla elevata non linearità del termine convettivo stesso. Per descriverla, in questa sezione si fa riferimento ad un generico problema monodimensionale di advezione-diffusione. Tale problema è descritto da

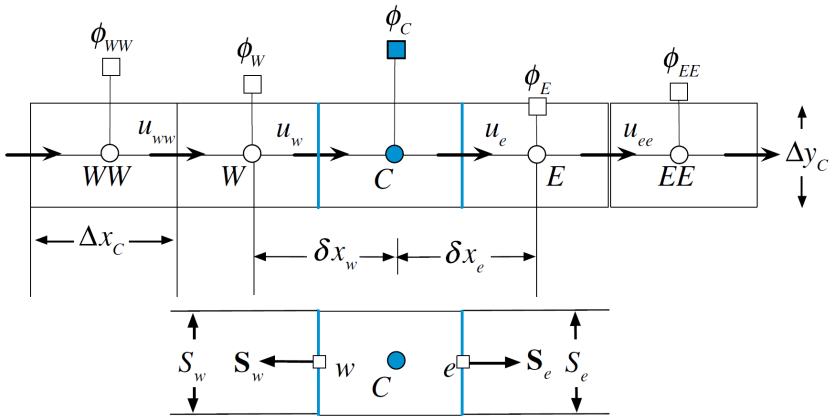


Figura 3.7: Griglia cartesiana nel problema di advezione-diffusione [24].

un'equazione di trasporto definita nel seguente modo:

$$\int_{V_C} \left[\nabla \cdot (\rho \mathbf{U}\phi) - \nabla \cdot (\Gamma^{\phi} \nabla \phi)\right] dV = 0 \tag{3.29}$$

Applicando il teorema di Gauss e riscrivendo l'integrale superficiale come somma dei flussi attraverso le facce:

$$\sum_{f \sim nb(C)} \left(\rho U \phi \mathbf{i} - \Gamma^{\phi} \frac{d\phi}{dx}\mathbf{i}\right)_f \cdot \mathbf{S}_f = 0 \tag{3.30}$$

ed esplicitando la somma delle due facce:

$$\left[(\rho U \Delta y \phi)_e - \left(\Gamma^\phi \frac{d\phi}{dx}\Delta y\right)_e\right] - \left[(\rho U \Delta y \phi)_w - \left(\Gamma^\phi \frac{d\phi}{dx}\Delta y\right)_w\right] = 0 \tag{3.31}$$

si ottiene l'equazione di trasporto discretizzata. Il valore della velocità alle facce delle celle è noto, le proprietà del fluido sono note e il gradiente di ϕ può essere discretizzato come spiegato nella sezione precedente (Eq.3.27). Gli unici valori non noti sono i valori che la proprietà ϕ assume in corrispondenza delle facce e,w rispetto ai valori presenti nei centroidi delle celle adiacenti ϕ_E, ϕ_C, ϕ_W. Esistono diversi schemi di discretizzazione discussi in letteratura, noti anche come schemi di advezione, che introducono diversi metodi per definire tali valori. In questo lavoro sono stati utilizzati degli schemi del primo e del secondo ordine; pertanto, vengono riportati in seguito alcuni cenni riguardo gli schemi *upwind*.

Schemi upwind

Secondo i metodi upwind, il valore di ϕ sulla faccia è funzione del valore memorizzato nel centroide delle celle che precedono il moto del fluido. Tale valore verrà dunque definito nel seguente modo:

$$\phi_e = \begin{cases} \phi_C & : \ \dot{m}_e > 0 \\ \phi_E & : \ \dot{m}_e < 0 \end{cases} \qquad and \qquad \phi_w = \begin{cases} \phi_C & : \ \dot{m}_w > 0 \\ \phi_W & : \ \dot{m}_w < 0 \end{cases} \tag{3.32}$$

dove $\dot{m}_e$ e $\dot{m}_w$ rappresentano le portate massiche alle facce e,w; esse sono definite come:

$$\dot{m}_e = (\rho \mathbf{U} \cdot \mathbf{S})_e = (\rho U S)_e = (\rho U \Delta y)_e \tag{3.33}$$

$$\dot{m}_w = -(\rho \mathbf{U} \cdot \mathbf{S})_w = -(\rho U S)_w = -(\rho U \Delta y)_w \tag{3.34}$$

Pertanto, si può esprimere il flusso convettivo alla faccia e nel seguente modo:

$$\begin{gathered} \dot{m}_e \phi_e = max\{\dot{m}_e,0\}\phi_C - max\{-\dot{m}_e,0\}\phi_E = \\ = FluxC_e^{Conv}\phi_C + FluxF_e^{Conv}\phi_E + FluxV_e^{Conv} \end{gathered} \tag{3.35}$$

Il procedimento è analogo per il flusso convettivo alla faccia w. Applicando uno schema di discretizzazione lineare al termine diffusivo (Eq.3.28) ed uno schema upwind del primo ordine al termine convettivo (Eq.3.35) nel problema di advezione-diffusione (Eq.3.29), si ottiene la seguente equazione:

$$\begin{gathered} (FluxC_e^{Conv} + FluxC_e^{Diff} + FluxC_w^{Conv} + FluxC_w^{Diff})\phi_C + \\ (FluxF_e^{Conv} + FluxF_e^{Diff})\phi_E + \\ (FluxF_w^{Conv} + FluxF_w^{Diff})\phi_W = 0 \end{gathered} \tag{3.36}$$

che può essere riscritta nella seguente forma canonica:

$$a_C\phi_C + a_E\phi_E + a_W\phi_W = 0 \tag{3.37}$$

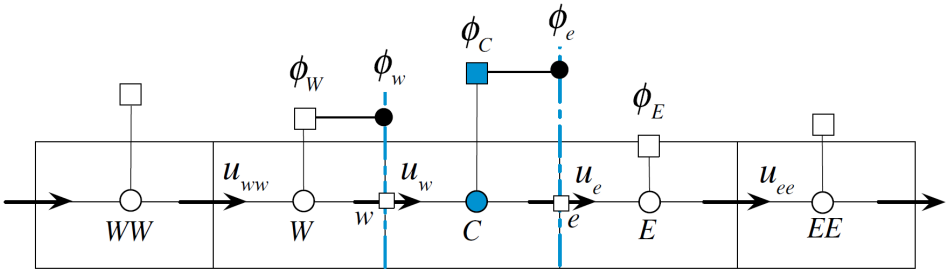


Figura 3.8: Profilo dello schema upwind [24].

Altri schemi

Gli schemi di discretizzazione sono caratterizzati da alcune proprietà computazionali, come la limitatezza, la stabilità e l'accuratezza della soluzione. Metodi del secondo ordine, come il *central differencing scheme* (CDS) forniscono una migliore accuratezza della soluzione, riducendo l'errore ad ogni iterazione successiva ad una maggiore velocità rispetto ai metodi del primo ordine. Tuttavia, alcuni metodi di ordine superiore al primo sono affetti da problemi di stabilità e limitatezza. Tali metodi possono introdurre errori numerici tra le varie iterazioni quando il fenomeno convettivo presenta un contributo importante nel sistema da studiare, cioè quando il Pe_{cella}:

$$Pe_{cella} = \frac{\rho U \delta_x}{\Gamma^\phi} \tag{3.38}$$

che misura il rapporto tra la velocità del trasporto convettivo e la velocità del trasporto diffusivo, presenta valori elevati nel sistema. Spesso gli schemi di ordine superiore al primo presentano stabilità e limitatezza solo per determinati valori di Pe_{cella}, pertanto non sono sempre adoperabili.
Tra gli schemi di ordine superiore al primo si possono citare anche il *second order upwind* ed il metodo QUICK, che fornisce una accuratezza del terzo ordine [24].

3.4 Discretizzazione temporale

La discretizzazione temporale può essere descritta in modo analogo alla discretizzazione del termine convettivo. La generica equazione di trasporto (Eq.(3.1))

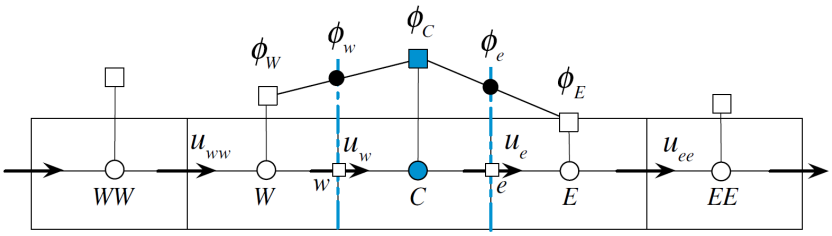


Figura 3.9: Profilo dello schema CDS (schema del secondo ordine) [24].

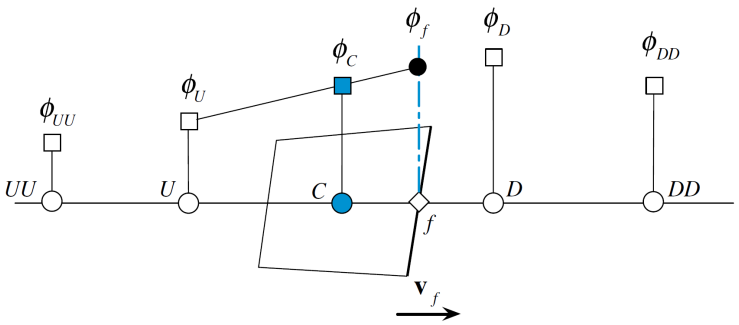


Figura 3.10: Profilo dello schema *second order upwind* (schema del secondo ordine) [24].

può essere riscritta raccogliendo i termini dipendenti dal tempo e quelli dipendenti dallo spazio:

$$\frac{\partial(\rho\phi)}{\partial t} + \mathcal{L}(\phi) = 0 \tag{3.39}$$

dove $\mathcal{L}(\phi)$ rappresenta l'operatore di discretizzazione spaziale, che contiene il termine diffusivo, quello convettivo e quello sorgente.
Integrando sulla cella C si ottiene:

$$\int_{V_C} \frac{\partial(\rho\phi)}{\partial t} dV + \int_{V_C} \mathcal{L}(\phi) dV = 0 \tag{3.40}$$

$$\frac{\partial(\rho\phi)}{\partial t} V_C + \mathcal{L}(\phi_C^t) = 0 \tag{3.41}$$

L'operatore di discretizzazione spaziale può essere scritto in forma algebrica come:

$$\mathcal{L}(\phi_C^t) = a_C \phi_C^t + \sum_{f \sim nb(C)} a_F \phi_F^t - b_C \tag{3.42}$$

dove ϕ_C^t e ϕ_F^t sono i coefficienti di discretizzazione spaziale al tempo t.
Integrando l'Eq.(3.41) nel tempo, si ottiene la forma semi-discreta dell'equazione

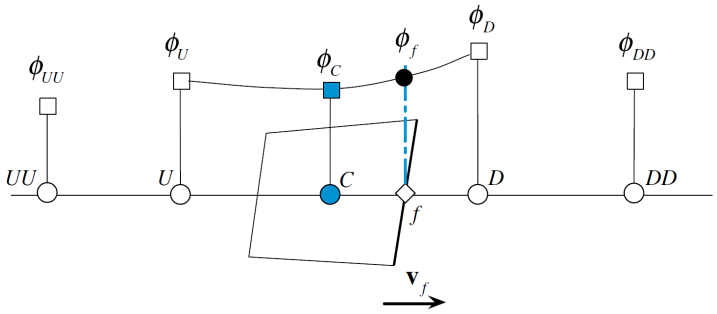


Figura 3.11: Profilo dello schema QUICK (schema del terzo ordine) [24].

di trasporto transitoria:

$$\int_{t-\Delta t/2}^{t+\Delta t/2} \frac{\partial(\rho\phi_C)}{\partial t} V_C dt + \int_{t-\Delta t/2}^{t+\Delta t/2} \mathcal{L}(\phi_C) dt = 0 \tag{3.43}$$

$$\frac{(\rho\phi_C)^{t+\Delta t/2} - (\rho\phi_C)^{t-\Delta t/2}}{\Delta t} V_C + \mathcal{L}(\phi_C^t) = 0 \tag{3.44}$$

L'equazione è completamente discretizzata se si sceglie un profilo di interpolazione per esprimere i valori di ϕ in $(t-\Delta t/2)$ e $(t+\Delta t/2)$ in funzione dei valori al tempo (t) e $(t-\Delta t)$. Esistono molteplici metodi di interpolazione trattati in letteratura, ma nel presente lavoro è stato adottato il metodo di Eulero implicito, un metodo del primo ordine incondizionatamente stabile.
Tale metodo prevede che:

$$(\rho\phi_C)^{t+\Delta t/2} = (\rho\phi_C)^t \tag{3.45}$$

$$(\rho\phi_C)^{t-\Delta t/2} = (\rho\phi_C)^{t-\Delta t} \tag{3.46}$$

Applicando il metodo all'Eq.(3.44) si ottiene la forma totalmente discretizzata dell'equazione di trasporto:

$$\frac{(\rho\phi_C)^t - (\rho\phi_C)^{t-\Delta t}}{\Delta t} V_C + \mathcal{L}(\phi_C^t) = 0 \tag{3.47}$$

3.5 OpenFOAM

`OpenFOAM` [26] è un software di fluidodinamica computazionale open-source e programmato in linguaggio `C++`. Esso presenta una vasta gamma di *solver* che permette di risolvere le equazioni di trasporto in numerosi campi di applicazione ingegneristica. `OpenFOAM` mette a disposizione, inoltre, diversi *tools* utili nelle fasi di pre-processing e post-processing dell'analisi CFD.
In questo paragrafo verrà fornita una breve descrizione del funzionamento del software e dei principali strumenti utilizzati per questo lavoro.

3.5.1 Impostazione dei *foamCases*

Ogni simulazione che viene lanciata su `OpenFOAM` è costituita da un *foamCase*, la cui struttura è definita dal seguente insieme di cartelle:

- `0`: rappresenta le condizioni iniziali della simulazione. Contiene una serie di file nei quali sono descritte le condizioni iniziali e di bordo per ogni campo che si vuole simulare;

- `constant`: contiene i file che definiscono le proprietà del fluido e il tipo di simulazione che verrà svolta. Durante il meshing della geometria, verrà creata al suo interno la cartella polyMesh, nella quale sono contenuti i file di topologia della mesh;

- `system`: i file contenuti in questa cartella definiscono il solver adottato, gli schemi di discretizzazione utilizzati ed altre specifiche relative alle iterazioni del solver;

- `time_directories`: vengono create dal software e contengono la soluzione della simulazione, divisa per ogni campo simulato. È possibile decidere la frequenza con il quale i risultati vengono salvati in queste cartelle.

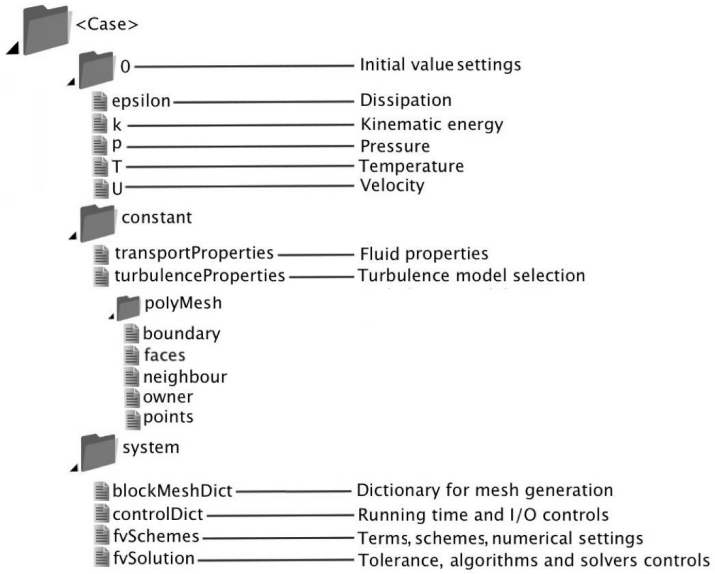


Figura 3.12: Struttura di un *foamCase*.

3.5.2 Meshing tools

Per definire la mesh computazionale sono utilizzati principalmente tre strumenti forniti da `OpenFOAM`:

- `blockMesh`: questo tool ha il compito di definire il dominio cubico all'interno del quale verrà posizionata la mesh effettiva del mezzo poroso. Configurando il parametro che rappresenta il numero di celle per lato, è possibile definire il numero di celle totali di cui sarà composto il dominio, definendo di conseguenza la risoluzione della mesh;

- `snappyHexMesh`: si occupa di posizionare il mezzo poroso (fornito sotto forma di file `.stl`) al centro della mesh cubica generata dal `blockMesh`. Il tool localizza i bordi della geometria ed infittisce la mesh nelle celle che intersecano tali bordi; il processo verrà in seguito ripetuto fino al raggiungimento del grado di discretizzazione desiderato;

- `checkMesh`: tool che analizza la mesh appena generata e ne valuta la qualità. Fornisce un file dettagliato in cui sono presenti i valori di alcuni importanti per la determinazione della qualità di una mesh, come la *skewness* o l'ortogonalità delle celle; se la mesh supera determinati limiti imposti a questi parametri, essa verrà definita inadatta poiché potrebbe portare ad avere dei risultati affetti da errore non trascurabile.

3.5.3 Solver

`OpenFOAM` fornisce agli utenti dei solver già strutturati e configurati, specifici per determinati casi studio. In questo lavoro le simulazioni del campo di moto sono state eseguite con il solver `interFoam`. `interFoam` è un *solver* al transitorio per sistemi multifase ed è basato sull'algoritmo PIMPLE, una combinazione degli algoritmi PISO (Pressure Implicit with Splitting of Operators) e SIMPLE (Semi-Implicit Method for Pressure Linked Equations).
L'algoritmo consiste nell'integrare l'algoritmo PISO (utilizzato per risolvere i problemi di *pressure-velocity coupling*) nell'algoritmo PIMPLE (utilizzato per risolvere le equazioni di Navier-Stokes) (Fig.3.13).

3.6 Risorse computazionali

Lo studio della fluidodinamica attraverso simulazioni numeriche permette di risparmiare numerose risorse in ambito sperimentale, pertanto il suo utilizzo è in costante aumento in molti campi ingegneristici. Tali simulazioni presentano un'accuratezza che dipende, oltre che dalla formulazione degli algoritmi numerici, anche

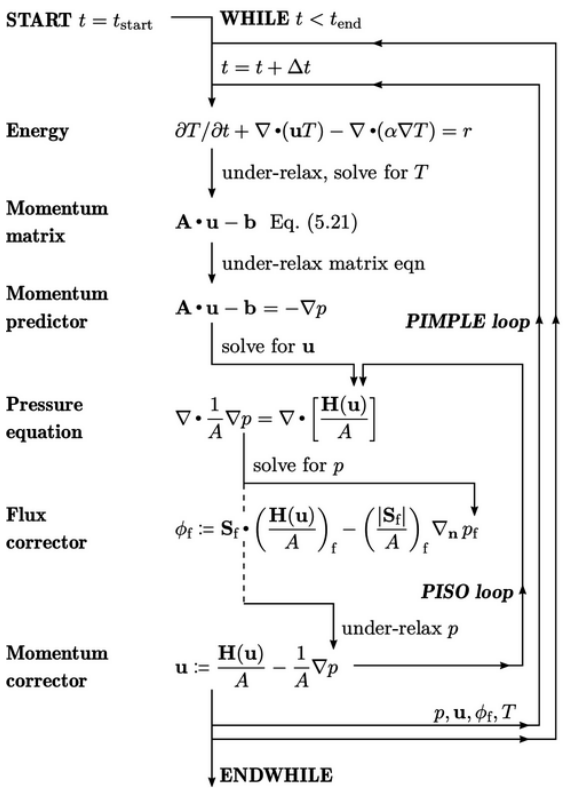


Figura 3.13: Schema dell'algoritmo PIMPLE [27].

dalla discretizzazione del dominio; tanto più è grande il numero di celle, tanto più diminuisce l'errore numerico. La capacità di calcolo necessaria per gestire l'intensa discretizzazione del dominio studiato in questo lavoro è stata fornita da HPC@Polito [28]. Tutte le simulazioni condotte nell'ambito di questo lavoro si sono svolte sul cluster di ateneo del Politecnico di Torino: legion@polito.it. I sistemi HPC, come quello utilizzato per questo lavoro, permettono di svolgere più attività in parallelo ed abbattere significativamente i tempi computazionali distribuendo il calcolo su più processori. In questo lavoro, tutte le simulazioni sono state condotte imponendo una parallelizzazione su 8 *cores* per le fasi di meshing (`snappyHexMesh`) e di risoluzione delle equazioni di governo (`interFoam`).

Capitolo 4

Setup delle simulazioni

In questo capitolo viene descritta l'impostazione delle simulazioni eseguite su `OpenFOAM`. In particolare sono discussi in dettaglio le geometrie create nella sezione 4.1, mentre i solver utilizzati e le condizioni al contorno implementate sono descritti nella sezione 4.2. Infine, la sezione 4.3 è dedicata al postprocessing risultati delle simulazioni.

4.1 Geometria

I mezzi porosi generati in questo lavoro sono stati ottenuti tramite il software open-source `Yade`, sulla base di una serie di parametri geometrici forniti come input. Il software `Yade` è un framework open-source per modelli numerici discreti [29] ed è stato utilizzato per creare delle geometrie monodisperse periodiche randomiche, costituite dunque da delle sfere di diametro fisso che possono attraversare i bordi del dominio computazionale grazie alla condizione di periodicità (Fig.4.1). Tramite l'utilizzo di `Python 3.11` è stato possibile redigere uno script in grado di leggere i parametri di input da un file `.yml` e di lanciare i comandi del framework `Yade` utili a creare le geometrie randomiche di interesse. I parametri utilizzati come input per lo script di creazione delle geometrie sono tre:

- il raggio delle sfere;
- il numero di sfere contenute in un lato del dominio computazionale;
- la varianza della distribuzione dei raggi delle sfere da generare.

In questo lavoro sono state create geometrie che contenessero delle sfere di raggio pari a 0.015 m costante (dunque è stata impostata una varianza del raggio pari a zero); il dominio computazionale presenta 9 sfere per lato. Il software `Yade` fornisce come output le dimensioni del dominio computazionale che conterrà la geometria,

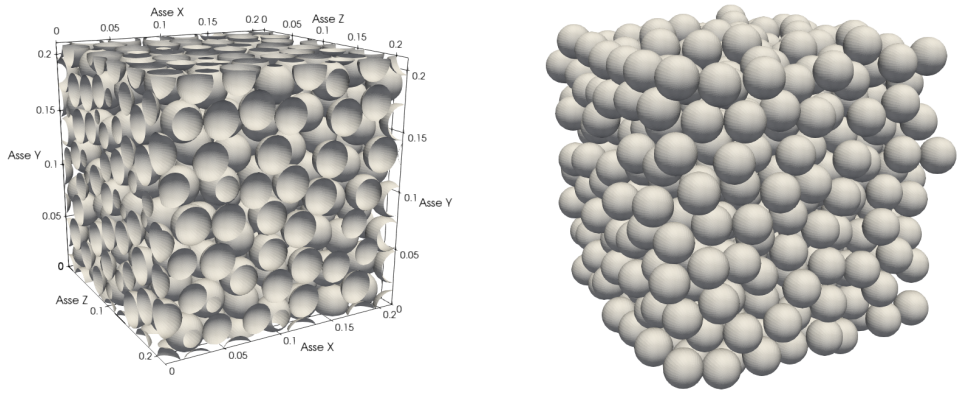


Figura 4.1: Esempio di geometria monodispersa periodica randomica.

il mezzo poroso generato sotto forma di file `.stl` (stereolitografico) e le coordinate di un punto interno al dominio ma esterno alla geometria (tale punto sarà richiesto da `OpenFOAM` per poter inizializzare correttamente il campo di moto dei due fluidi).

4.2 Condizioni al contorno e simulazioni effettuate

Le simulazioni sono state effettuate utilizzando il software di CFD open-source `OpenFOAMv2012`. Il dominio computazionale utilizzato in tutte le simulazioni presenta una forma cubica; le dimensioni di tale dominio dipendono dalla singola geometria presa in esame e vengono fornite dal software `Yade`. I fluidi considerati sono descritti nella seguente tabella:

Fluido	**Densità (kg m^{-3})**	**Viscosità cinematica (m^2 s^{-1})**
Aria	1	1.5×10^{-5}
Acqua	1000	0.89×10^{-6}

Tabella 4.1: Proprietà dei fluidi simulati

Il campo di moto è stato inizializzato imponendo un gradiente di pressione ΔP costante tra le patch di inlet e outlet, lungo l'asse x. Sulla superficie delle sfere è stata applicata la condizione di `zeroGradient` per la pressione, mentre le restanti patch sono state impostate con la condizione `cyclic`. Tale condizione permette al campo di moto di comportarsi in modo coerente con la geometria del mezzo poroso, garantendo la condizione di periodicità. Per quanto riguarda la velocità

U, è stata applicata la condizione di `noSlip` sulla superficie delle sfere, mentre le altre patch presentano la condizione `cyclic`. La distribuzione dei due fluidi all'interno del dominio è stata impostata nel seguente modo: inizialmente viene inizializzato tutto il dominio con aria. Successivamente, in base alla frazione d'acqua α_{water} desiderata, viene calcolata una specifica coordinata dell'asse x chiamata x_{water}. Infine, viene sostituita con acqua tutta la parte di dominio compresa tra le coordinate $x = 0$ (patch di `inlet`) e $x = x_{water}$.

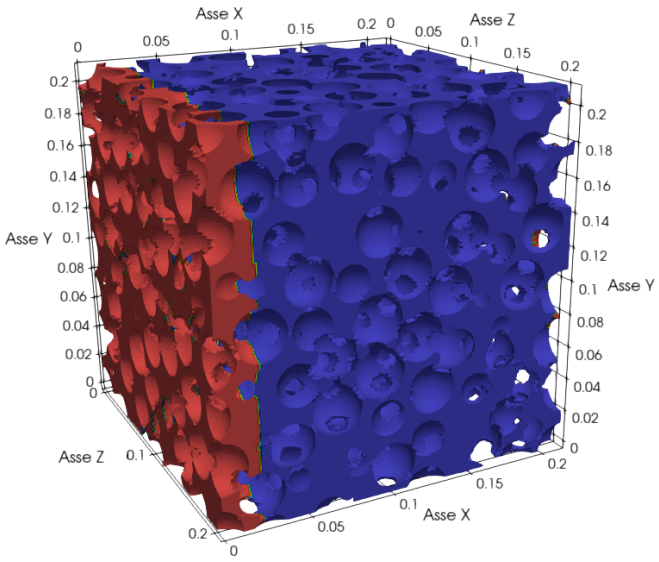


Figura 4.2: Distribuzione dei fluidi in un caso inizializzato con $\alpha_{water} = 0.1$.

Le simulazioni sono state condotte tramite il solver `interFoam`, adatto per simulazioni al transitorio di due fluidi incomprimibili e immiscibili.

4.2.1 Criterio di stop simulazione

Le simulazioni restituiscono come output, per ogni istante temporale simulato, i campi di pressione e di velocità dell'intero sistema e i valori di frazione d'acqua per ogni cella del dominio computazionale. Pertanto per ricavare la permeabilità del mezzo poroso, proprietà che dovrà essere predetta dal modello data-driven, viene utilizzata la legge di Darcy presentata nella sezione 2.1.2.
Il solver `interFoam` non è provvisto di un criterio di stop simulazione, in quanto esso lavora *al transitorio*. Pertanto, si è deciso di definire una condizione che permettesse di interrompere le simulazioni una volta che il sistema simulato raggiunge la cosiddetta condizione di *pseudo-steady state*. Il sistema oggetto d'esame di questo lavoro è caratterizzato da un flusso instabile, che non è in grado cioè

di raggiungere una condizione di flusso stazionario; tuttavia, dopo una prima fase di flusso transitorio, le proprietà del sistema oscillano indefinitamente all'infinito all'interno di un determinato range. Se tale range risulta essere più piccolo di un certo intervallo di tolleranza, scelto opportunamente tenendo conto degli ordini di grandezza del sistema, allora si dice che il flusso ha raggiunto una condizione di pseudo-stazionario.
È stato dunque prodotto uno script `python` che fosse in grado di valutare la media volumetrica della velocità del flusso simulato e di ordinare al solver di interrompere la simulazione nel caso in cui tale media raggiungesse la condizione di pseudo-stazionario e la mantenesse per un intervallo di tempo opportunamente definito a priori.

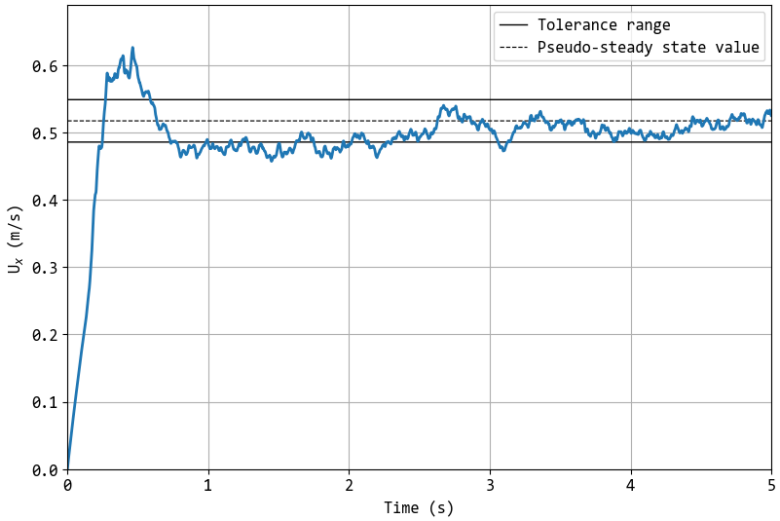


Figura 4.3: Esempio di simulazione che ha soddisfatto il criterio di stop per un intervallo Δt pari a 1,5 s.

4.2.2 Impostazione dataset

Come presentato nella sezione 2.2, al fine di ottenere un modello data-driven è necessario stabilire quali siano le feature caratteristiche del sistema. Nell'ambito di questo lavoro, il cui obiettivo è quello di ottenere la permeabilità delle due fasi nel mezzo poroso, è stato scelto di utilizzare come parametri di input al modello il gradiente di pressione $\Delta P/\Delta L$ e la frazione d'acqua α_{water}. In Tab.4.2 sono riportati i range di variazione dei parametri entro cui sono state realizzate le simulazioni utili all'ottenimento del dataset.

Parametro	Range
$\Delta P/\Delta L$	100 - 500 (Pa/m)
α_{water}	0 - 1 (-)

Tabella 4.2: Range di variazione dei parametri per la creazione del dataset.

4.3 PostProcessing

Il solver `interFoam` fornisce come output della simulazione un insieme di cartelle: per ogni istante temporale calcolato viene creata una cartella che contiene dei file che descrivono i campi di tutte le proprietà del fluido cella per cella. Inoltre, il solver produce anche un file *log* in cui sono contenute le informazioni relative allo stato della simulazione, anch'esse suddivise per ogni istante temporale computato. Il software `OpenFOAM` è provvisto di una serie di funzioni aggiuntive che possono essere importate ed eseguite ad ogni istante temporale; l'output di tali funzioni viene inserito nel file *log*, di cui si mostra un estratto in seguito:

```
deltaT = 0.00020031732
Time = 1.62217
ExecutionTime = 48126.01 s  ClockTime = 50056 s

volFieldValue volume_average write:
    volAverage(region0) of U = (0.072994511 -0.00095244031
       -0.0010709563)
    volAverage(region0) of p = 14.162586

surfaceFieldValue inlet_average write:
    areaAverage(inlet) of U = (0.068503545 0.002055348 0.0021982013)
    areaAverage(inlet) of p = 49.265721

surfaceFieldValue outlet_average write:
    areaAverage(outlet) of U = (0.068503545 0.002055348 0.0021982013)
    areaAverage(outlet) of p = -14.606675

magU_total  : (0.077792607 -0.011050494 6.125281e-05)
magU_water  : (0.046682204 -0.0023067338 -0.00040613534)
magU_air    : (0.10981688 -0.020051103 0.00054237051)
p_total     : 7.2830882
p_water     : 6.6365386
p_air       : 7.9486303
Courant Number mean: 0.018656458 max: 0.49931336
Interface Courant Number mean: 0.0039986446 max: 0.20715225
```

Listing 4.1: Esempio di file *log* di una simulazione.

Le funzioni di `OpenFOAM` sopraccitate sono definite *function objects* [30] e sono molto utili in fase di *postprocessing*. Le *function objects* utilizzate durante le simulazioni svolte in questo lavoro sono:

- `surfaceFieldValue`: fornisce delle operazioni utili a manipolare i dati relativi ai campi superficiali, che servono a valutare i flussi attraverso una *patch*. In questo lavoro, è stata utilizzata l'operazione `areaAverage` per calcolare la velocità media e la pressione in corrispondenza delle *patch* di `inlet` e di `outlet`.

- `volFieldValue`: function object che dispone di operazioni utili a manipolare i dati relativi alle proprietà volumetriche del sistema. In questo lavoro, è stata utilizzata l'operazione `volAverage` per valutare la velocità media e la pressione dell'intera fase fluida del sistema (definita `region0`).

- `coded`: strumento che permette di manipolare i dati dell'intero sistema attraverso delle operazioni *user-defined*, cioè interamente scritte in linguaggio `C++` dall'utente. In questo lavoro, è stata utilizzata questa function object per definire un'operazione che, tramite i campi di U, P e α_{water} simulati da `interFoam`, potesse calcolare le velocità e le pressioni delle singole fasi (aria e acqua).

Al fine di raccogliere i risultati in un'adeguata struttura dati, è stato elaborato uno script `python` in grado di leggere le informazioni rilevanti per questo studio dal file *log* di `interFoam`. In particolare, lo script crea un file `.csv` in cui sono presenti: il gradiente di pressione $\Delta P/\Delta L$ e la frazione d'acqua α_{water} al quale è stata condotta la simulazione; i valori della velocità media U del sistema lungo la direzione x di scorrimento del fluido, calcolati da `volFieldValue`; i valori delle velocità medie delle due fasi U_w, U_a lungo la direzione x, calcolati dalla function object `coded`.
Successivamente, grazie a un ulteriore script `python`, sono stati calcolati i valori pseudo-stazionari delle velocità citate precedentemente. Sfruttando le relazioni di Darcy presentate nella sezione 2.1.2, dai valori pseudo-stazionari delle velocità sono state ricavate anche le permeabilità relative delle due fasi; è stato realizzato infine un database contenente i valori delle velocità allo pseudo-stazionario e delle permeabilità relative di ogni simulazione, anch'esso sotto forma di file `.csv`.

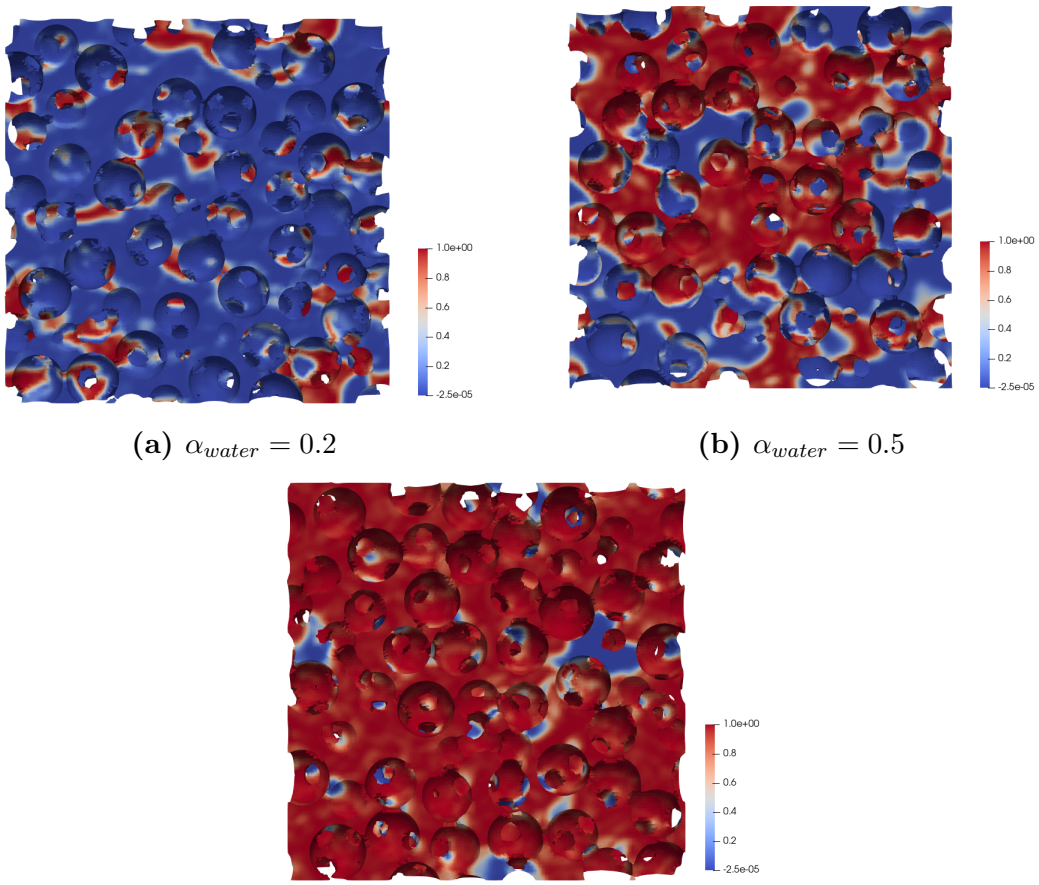
Capitolo 5

Risultati

In questo capitolo sono presentati i risultati dello studio condotto durante questo lavoro di tesi. Nella sezione 5.1 vengono discussi i risultati ottenuti dalle simulazioni di fluidodinamica computazionale, mentre nella sezione 5.2 vengono commentate le prestazioni del modello data-driven costruito tramite i risultati delle simulazioni di CFD.

5.1 Simulazioni di CFD

Le simulazioni sono state condotte tramite il software `OpenFOAM`, fornendo ad ogni geometria studiata dei valori iniziali di $\Delta P/\Delta L$ e di α_{water}. L'output di ogni simulazione è costituito dai campi di velocità U, pressione P e frazione d'acqua α_{water}, calcolati dal software `OpenFOAM` per ogni istante temporale simulato. La fase del transitorio del sistema, tuttavia, non è di interesse per questo studio; pertanto, verranno discussi soltanto i risultati relativi alle condizioni pseudo-stazionarie dei sistemi simulati. In Fig.5.1 sono mostrati degli esempi di casi che hanno raggiunto la condizione di pseudo-stazionario; in particolare, viene mostrata la distribuzione delle fasi una volta raggiunta tale condizione. I campi di velocità del fluido bifase $\mathbf{U}$, valutata da `interFoam` secondo la relazione 2.1.2, sono raffigurati in Fig.5.2. I risultati presentano dei campi di velocità molto omogenei; questo mostra ancora più chiaramente il raggiungimento di uno pseudo-stazionario. Durante la fase di post-processing, combinando le informazioni ottenute dalle simulazioni, è stato possibile calcolare le velocità delle singole fasi $\mathbf{U}_a$,$\mathbf{U}_w$ e il loro valore volumetrico medio. Successivamente, sfruttando le relazioni del modello di Darcy per sistemi bifase presentate nella sezione 2.1.2, sono stati calcolati i valori delle permeabilità relative delle due fasi κ_w,κ_a. Durante la campagna di simulazioni sono stati valutati diversi valori di frazione d'acqua α_{water} e ne è stata studiata la correlazione con le corrispondenti permeabilità relative. In Fig.5.3 è possibile osservare la correla-

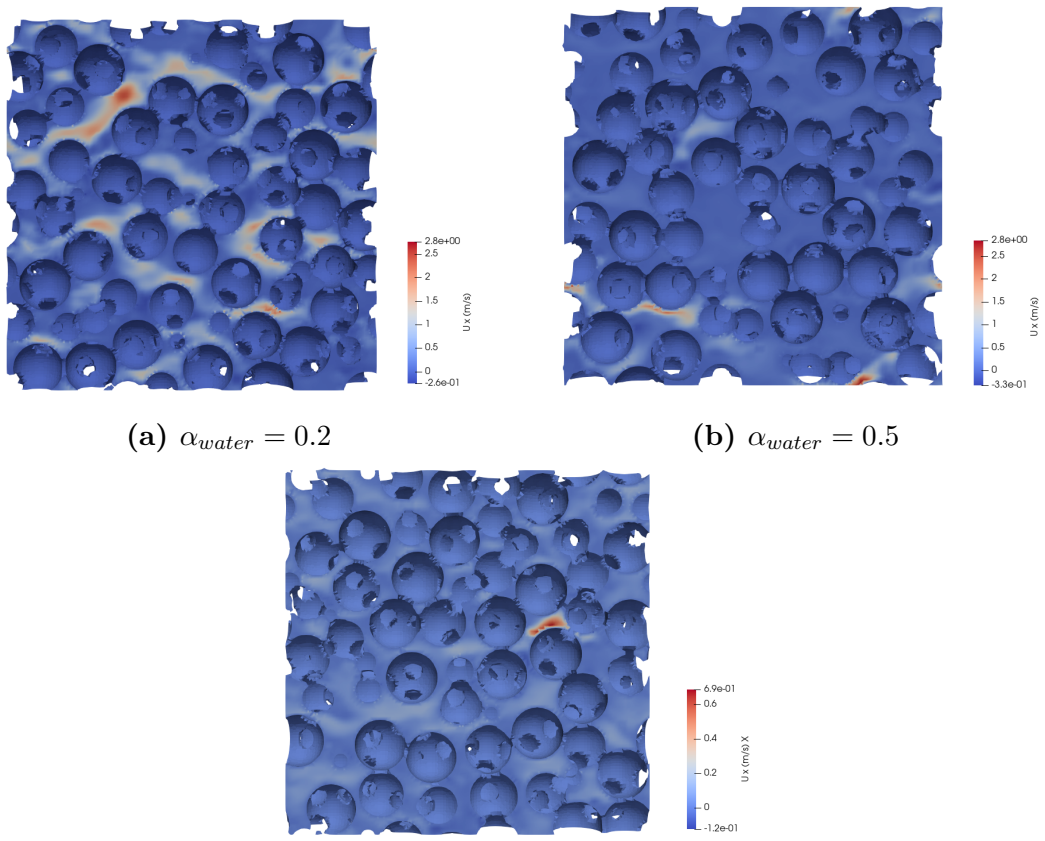


(a) $\alpha_{water} = 0.2$ **(b)** $\alpha_{water} = 0.5$

(c) $\alpha_{water} = 0.8$

Figura 5.1: Distribuzione delle fasi per diversi valori di α_{water}.

zione tra le permeabilità relative delle singole fasi e la frazione volumetrica d'acqua del flusso bifase. La permeabilità relativa dell'aria κ_a presenta un comportamento coerente con il modello di Darcy bifase (Fig.2.1), decrescendo all'aumentare della frazione d'acqua del sistema. La permeabilità dell'acqua κ_w, invece, mostra un comportamento anomalo, specialmente per valori di α_{water} inferiori a 0.5: i risultati non presentano un andamento monotono crescente all'aumentare di α_{water}; inoltre, per frazioni d'acqua inferiori a 0.3, i risultati presentano dei valori maggiori rispetto alla condizione di fluido monofase in cui è presente solo acqua ($\alpha_{water} = 1$). Facendo riferimento alla Fig.5.1a, è possibile notare che quando la frazione d'acqua è molto bassa il fluido bifase è costituito da una fase continua (gassosa) in cui sono presenti delle gocce d'acqua di dimensione ridotta, quasi in forma dispersa; questo



(a) $\alpha_{water} = 0.2$ **(b)** $\alpha_{water} = 0.5$

(c) $\alpha_{water} = 0.8$

Figura 5.2: Campi di velocità (componente x) per diversi valori di α_{water}.

fa sì che il solver `interFoam` calcoli erroneamente in alcune zone dei valori puntuali di $\mathbf{U}_w$ elevati. Nonostante queste zone critiche possano non essere numerose, il loro contributo risulta significativo nel calcolo della velocità volumetrica media della fase liquida, portando a valori non coerenti con il comportamento atteso del sistema. Una soluzione che si può adottare è quella di infittire maggiormente la mesh computazionale del mezzo poroso; tuttavia, ciò comporterebbe un forte aumento dei tempi di simulazione e dei costi computazionali. Adottare questo tipo di soluzioni rende ancora più evidente che uno studio più approfondito sull'accoppiamento di simulazioni di fluidodinamica computazionale alla costruzione di modelli data-driven potrebbe agevolare in maniera concreta lo svolgimento degli studi futuri su questo lavoro.

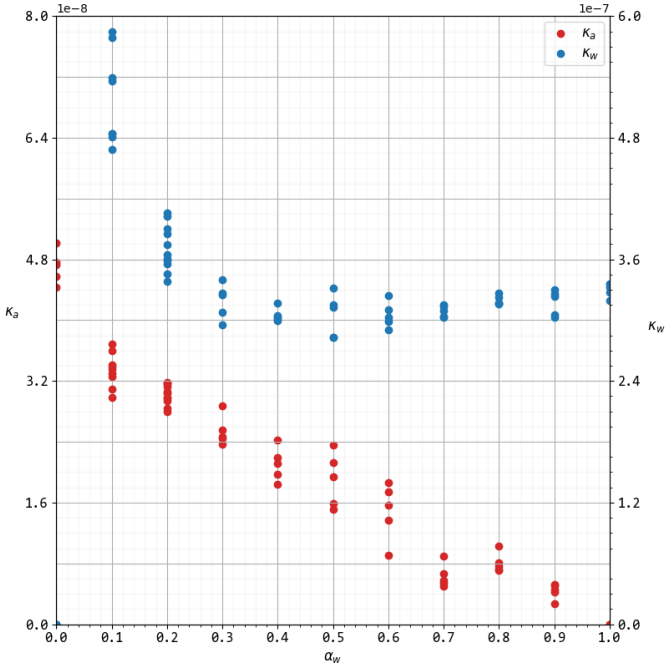


Figura 5.3: Permeabilità relative delle due fasi al variare di α_{water}. In rosso è rappresentata la permeabilità dell'aria κ_a, mentre in blu è mostrata quella dell'acqua κ_w.

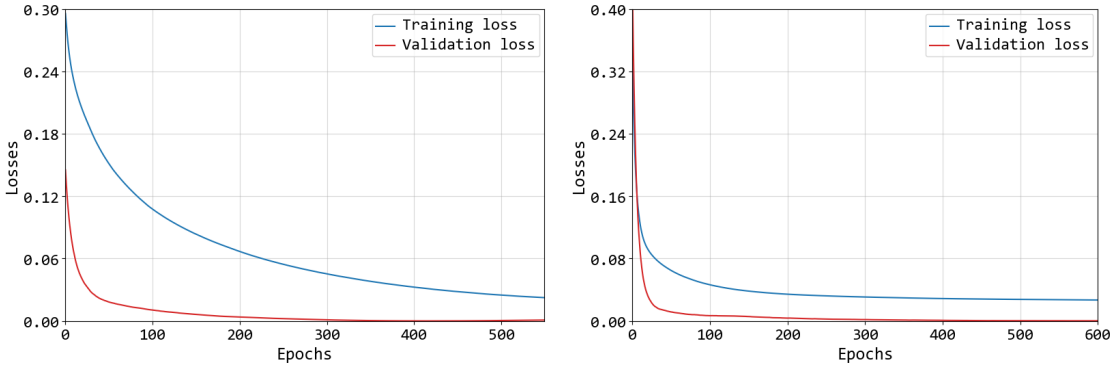
5.2 Modello data-driven

Conclusa la raccolta dei risultati dalle simulazioni, è stato creato un dataset costituito da 23 casi: all'interno del dataset il gradiente di pressione varia tra 100 Pa/m e 500 Pa/m, mentre la frazione d'acqua varia tra 0 e 1. Successivamente, è stata investigata la possibilità di impiegare un modello surrogato per predire i valori di permeabilità relativa delle due fasi κ_a,κ_w conoscendo solo i valori del gradiente di pressione $\Delta P/\Delta L$ e della frazione d'acqua α_{water} del sistema. A tale scopo, sono stati allenati due modelli di tipo ANN (artificial neural network), descritti nella sezione 2.2.3; entrambe le reti sono caratterizzate dalla presenza di due hidden layer omogenei. Nella seguente tabella sono mostrati i parametri utilizzati per costruire i due modelli:

Predizione	**# neuroni per layer**	**learning rate**	**batch size**	**epochs**
κ_a	10	5×10^{-4}	1	550
κ_w	10	1×10^{-3}	1	600

Tabella 5.1: Parametri di allenamento delle reti neurali.

In Fig.5.4 sono mostrate le *loss curves* delle due reti, rappresentative dell'allenamento eseguito con i parametri descritti in Tab.5.1. In entrambi i modelli è stata utilizzata la funzione di attivazione ReLU (Fig.2.3).

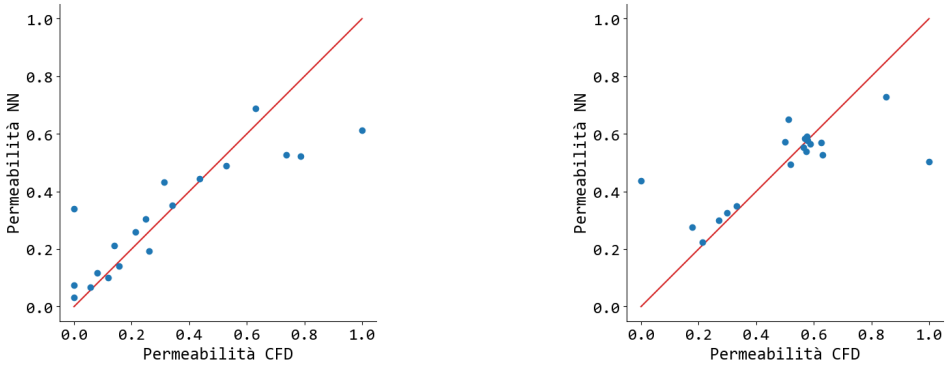


(a) Predizione della permeabilità relativa dell'aria κ_a.

(b) Predizione della permeabilità relativa dell'acqua κ_w.

Figura 5.4: *Loss curves* dell'allenamento dei modelli.

Le feature e gli output sono stati normalizzati per far sì che assumessero valori compresi tra 0 e 1, sfruttando le tecniche descritte nella sezione 2.2.1. Sono stati prodotti dei diagrammi di parità (Fig.5.5) al fine di valutare le prestazioni delle reti neurali nella predizione delle permeabilità relative.



(a) Predizione della permeabilità relativa dell'aria κ_a.

(b) Predizione della permeabilità relativa dell'acqua κ_w.

Figura 5.5: Diagrammi di parità della permeabilità relativa delle due fasi.

Per valutare le prestazioni dei modelli è stata utilizzata la metrica del *root mean square error* (RMSE), definito come la radice quadrata del MSE (Eq.2.26). Il modello di predizione della permeabilità relativa dell'aria presenta un valore di RMSE = 0.147, mentre quello che si occupa della permeabilità relativa dell'acqua

presenta un valore di RMSE = 0.181.
I valori di questa metrica mostrano che i modelli costruiti non sono ancora abbastanza accurati; questo è dovuto principalmente alle dimensioni estremamente ridotte del dataset. La soluzione principale consisterà dunque nell'estensione della campagna di simulazioni di fluidodinamica computazionale a dei casi caratterizzati da dei nuovi valori di gradiente di pressione e di frazione d'acqua. Tuttavia, i risultati hanno mostrato che è possibile valutare il futuro impiego dei modelli surrogati per far fronte agli elevati costi computazionali delle simulazioni di CFD sui flussi bifase che attraversano un mezzo poroso.

Capitolo 6

Conclusioni

Questo lavoro di tesi costituisce uno studio preliminare sul comportamento dei fluidi bifase in reattori trickle-bed per biometanazione. L'obiettivo dello studio condotto durante questo lavoro è quello di poter sfruttare i modelli alla microscala per ricavare dei parametri che descrivano il comportamento di un fluido bifase che attraversa un mezzo poroso, elemento costituente del letto del reattore. In questa fase preliminare sono state condotte delle simulazioni di fluidodinamica computazionale su un sistema non reattivo e costituito da aria e acqua che scorrono attraverso il mezzo poroso. Le fasi successive di questo studio saranno costituite dall'implementazione della simulazione della reazione 1.1: sarà necessario dunque sostituire la fase gassosa con una miscela di idrogeno e anidride carbonica in composizione stechiometrica. Inoltre, oltre allo studio del trasporto della quantità di moto del sistema, sarà cruciale investigare sul trasporto di materia e sullo scambio di calore che avverranno a causa dello svolgersi della reazione. Gli ultimi step dello studio riguarderanno invece l'accoppiamento delle simulazioni alla microscala (che descrivono il comportamento del fluido a livello locale, in una frazione elementare del letto del reattore) con i modelli alla macroscala, che descrivono invece il comportamento del fluido lungo tutto il reattore: le fasi di studio della microscala sono stati impostati in modo tale da ricavare come proprietà target la permeabilità del mezzo poroso, poiché essa è una proprietà che non dipende dalle dimensioni del sistema. Sarà possibile, dunque, sfruttare le simulazioni alla microscala per valutare la permeabilità di un dato mezzo poroso; la permeabilità ricavata, infine, sarà fornita come input ai modelli alla macroscala che permetteranno di descrivere in maniera più accurata il comportamento del fluido attraverso l'intero letto del reattore. Come descritto nella sezione 1.3.3, ogni fase di questo studio è ostacolata da degli elevati costi computazionali che risultano essere limitanti sia in termini di risorse che di tempo. Pertanto, anche durante le fasi successive sarà fondamentale valutare la possibilità di affiancare le simulazioni di fluidodinamica computazionale alla costruzione di opportuni modelli data-driven; l'affiancamento permetterebbe

di estendere lo studio a molti più casi senza appesantire in maniera significativa il workflow.
In questo lavoro è stato utilizzato il software open-source `OpenFOAM` per condurre una campagna di simulazioni su dei mezzi porosi attraversati da un fluido bifase (aria - acqua); i parametri di input studiati durante la campagna sono stati il gradiente di pressione $\Delta P/\Delta L$ e la frazione d'acqua α_{water}, mentre l'output delle simulazioni è costituito dal campo di velocità medio del sistema $\mathbf{U}$ e dai campi di velocità delle singole fasi $\mathbf{U}_a$,$\mathbf{U}_w$. Le geometrie dei mezzi porosi sono state generate in maniera randomica e periodica utilizzando il utilizzato il software open-source `Yade`. Il post-processing dei risultati è stato svolto grazie a degli script stilati in linguaggio `Python` e ha permesso di utilizzare i campi di velocità prodotti dalle simulazioni per ricavare i valori di permeabilità relativa delle due fasi κ_a,κ_w. La campagna di simulazioni di fluidodinamica computazionale ha prodotto un dataset che è stato utilizzato per costruire un modello di regressione basato sulla struttura MLP (MultiLayer Perceptron) utilizzando gli strumenti forniti dalle librerie `Python` `scikit-learn` e `Keras`.
Per risolvere le equazioni presentate nella sezione 2.1 è stato utilizzato il solver `interFoam` di `OpenFOAM`. Si è investigata la corrispondenza tra i risultati delle simulazioni e il modello di Darcy per sistemi multifase, presentato nella sezione 2.1.2. I risultati hanno mostrato che le simulazioni di fluidodinamica computazionale sono state in grado di rappresentare la permeabilità relativa dell'aria κ_a in maniera coerente col modello di Darcy per qualsiasi valore di α_{water}; la permeabilità dell'acqua κ_w, invece, presenta dei risultati molto rumorosi e poco coerenti col modello di Darcy per valori di α_{water} inferiori a 0.5. Questo può essere dovuto alla scarsa capacità di `interFoam` di gestire le zone in cui sono presenti delle gocce d'acqua di dimensioni molto ridotte; la loro presenza porta ad avere valori puntuali di $\mathbf{U}_w$ molto elevati che influiscono significativamente sul calcolo della velocità media dell'acqua nel sistema. Una delle soluzioni che si possono adottare per far fronte a questa anomalia consiste nell'infittire maggiormente la mesh computazionale nei casi in cui si studiano i valori di α_{water} che portano ai risultati critici, anche se questo causerebbe un aumento esponenziale dei tempi di simulazione.
Sono stati allenati due modelli di tipo ANN (artificial neural network) caratterizzati dalla presenza di due hidden layer. I risultati hanno dimostrato che il dataset prodotto dalla campagna di simulazioni di fluidodinamica computazionale non presenta delle dimensioni sufficienti ad allenare un modello efficiente per la predizione delle permeabilità relative. Tuttavia, gli stessi risultati mostrano che l'impiego dei modelli surrogati potrà essere fondamentale per far fronte agli elevati costi computazionali delle simulazioni di CFD sui flussi bifase che attraversano un mezzo poroso. Uno degli obiettivi principali dei lavori futuri potrebbe essere dunque quello di estendere la campagna di simulazioni ad un range più ampio di valori esplorati di gradiente di pressione e frazione d'acqua.

Bibliografia

[1] *IPCC glossary website*. URL: https://www.ipcc.ch/sr15/chapter/glossary/ (visitato il 08/09/2024) (cit. a p. 1).

[2] United Nations Environment Programme. *Emissions Gap Report 2023: Broken Record – Temperatures hit new highs, yet world fails to cut emissions (again)*. United Nations Environment Programme, nov. 2023. ISBN: 9789280740981. DOI: 10.59117/20.500.11822/43922. URL: http://dx.doi.org/10.59117/20.500.11822/43922 (cit. a p. 1).

[3] Amila Ajdinovic, Azrudin Husika e Sanjin Avdic. «Analysis of application of "Power to Gas" technology in Bosnia and Herzegovina». In: *TEM Journal* (ott. 2016). DOI: 10.18421/TEM54-17 (cit. a p. 3).

[4] Stefan Rönsch et al. «Review on methanation – From fundamentals to current projects». In: *Fuel* 166 (feb. 2016), pp. 276–296. ISSN: 0016-2361. DOI: 10.1016/j.fuel.2015.10.111. URL: http://dx.doi.org/10.1016/j.fuel.2015.10.111 (cit. a p. 3).

[5] Bernhard Lecker et al. «Biological hydrogen methanation – A review». In: *Bioresource Technology* 245 (dic. 2017), pp. 1220–1228. ISSN: 0960-8524. DOI: 10.1016/j.biortech.2017.08.176. URL: http://dx.doi.org/10.1016/j.biortech.2017.08.176 (cit. a p. 4).

[6] Yan Rafrafi, Léa Laguillaumie e Claire Dumas. «Biological Methanation of H2 and CO2 with Mixed Cultures: Current Advances, Hurdles and Challenges». In: *Waste and Biomass Valorization* 12.10 (nov. 2020), pp. 5259–5282. ISSN: 1877-265X. DOI: 10.1007/s12649-020-01283-z. URL: http://dx.doi.org/10.1007/s12649-020-01283-z (cit. a p. 4).

[7] Solagro., E&E Consultant e Hespul. *Etude sur l'hydrogène et la méthanation comme procédé de valorisation de l'électricité excédentaire*. ADEME, GrDF, GRTgaz, 2014 (cit. a p. 4).

[8] Davis Rusmanis et al. «Biological hydrogen methanation systems – an overview of design and efficiency». In: *Bioengineered* 10.1 (gen. 2019), pp. 604–634. ISSN: 2165-5987. DOI: 10.1080/21655979.2019.1684607. URL: http://dx.doi.org/10.1080/21655979.2019.1684607 (cit. alle pp. 4, 6).

[9] Ruggero Bellini et al. «Biological Aspects, Advancements and Techno-Economical Evaluation of Biological Methanation for the Recycling and Valorization of CO2». In: *Energies* 15.11 (giu. 2022), p. 4064. ISSN: 1996-1073. DOI: 10.3390/en15114064. URL: http://dx.doi.org/10.3390/en15114064 (cit. a p. 5).

[10] M.A. Voelklein, Davis Rusmanis e J.D. Murphy. «Biological methanation: Strategies for in-situ and ex-situ upgrading in anaerobic digestion». In: *Applied Energy* 235 (feb. 2019), pp. 1061–1071. ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2018.11.006. URL: http://dx.doi.org/10.1016/j.apenergy.2018.11.006 (cit. a p. 5).

[11] Hao Jiang et al. «Biological methanation of H2 and CO2 in a continuous stirred tank reactor». In: *Journal of Cleaner Production* 370 (ott. 2022), p. 133518. ISSN: 0959-6526. DOI: 10.1016/j.jclepro.2022.133518. URL: http://dx.doi.org/10.1016/j.jclepro.2022.133518 (cit. a p. 6).

[12] Amita Jacob Guneratnam et al. «Study of the performance of a thermophilic biological methanation system». In: *Bioresource Technology* 225 (feb. 2017), pp. 308–315. ISSN: 0960-8524. DOI: 10.1016/j.biortech.2016.11.066. URL: http://dx.doi.org/10.1016/j.biortech.2016.11.066 (cit. a p. 6).

[13] Léa Laguillaumie et al. «Stability of ex situ biological methanation of H2/CO2 with a mixed microbial culture in a pilot scale bubble column reactor». In: *Bioresource Technology* 354 (giu. 2022), p. 127180. ISSN: 0960-8524. DOI: 10.1016/j.biortech.2022.127180. URL: http://dx.doi.org/10.1016/j.biortech.2022.127180 (cit. a p. 7).

[14] Savvas Savvas et al. «Biological methanation of CO2 in a novel biofilm plug-flow reactor: A high rate and low parasitic energy process». In: *Applied Energy* 202 (set. 2017), pp. 238–247. ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2017.05.134. URL: http://dx.doi.org/10.1016/j.apenergy.2017.05.134 (cit. alle pp. 7, 8).

[15] Carolina Feickert Fenske et al. «Biogas upgrading in a pilot-scale trickle bed reactor – Long-term biological methanation under real application conditions». In: *Bioresource Technology* 376 (mag. 2023), p. 128868. ISSN: 0960-8524. DOI: 10.1016/j.biortech.2023.128868. URL: http://dx.doi.org/10.1016/j.biortech.2023.128868 (cit. a p. 8).

[16] Carolina Feickert Fenske, Dietmar Strübing e Konrad Koch. «Biological methanation in trickle bed reactors - a critical review». In: *Bioresource Technology* 385 (ott. 2023), p. 129383. ISSN: 0960-8524. DOI: 10.1016/j.biortech.2023.129383. URL: http://dx.doi.org/10.1016/j.biortech.2023.129383 (cit. a p. 9).

[17] Simon Markthaler et al. «Numerical simulation of trickle bed reactors for biological methanation». In: *Chemical Engineering Science* 226 (nov. 2020), p. 115847. ISSN: 0009-2509. DOI: 10.1016/j.ces.2020.115847. URL: http://dx.doi.org/10.1016/j.ces.2020.115847 (cit. a p. 9).

[18] Santiago Márquez Damián. *Description and utilization of interFoam multiphase solver*. URL: https://www.cfdyna.com/Home/OpenFOAM/of_Tut_Web/of_Suites_Description.pdf (visitato il 05/11/2024) (cit. a p. 12).

[19] E. Castillo. *Relative Permeability Hysteresis in Porous Media*. 2016. URL: https://api.semanticscholar.org/CorpusID:115146800 (cit. a p. 14).

[20] Sergey Ioffe e Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. 2015. DOI: 10.48550/ARXIV.1502.03167. URL: https://arxiv.org/abs/1502.03167 (cit. a p. 16).

[21] F. Pedregosa et al. «Scikit-learn: Machine Learning in Python». In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830 (cit. a p. 18).

[22] François Chollet et al. *Keras*. https://keras.io. 2015 (cit. a p. 18).

[23] Agnese Marcato. «Accoppiamento di CFD e Machine Learning: due casi studio nell'ingegneria di processo= Coupling CFD and Machine Learning: two case studies in process engineering». Tesi di dott. Politecnico di Torino, 2019 (cit. alle pp. 19–21).

[24] F. Moukalled, L. Mangani e M. Darwish. *The Finite Volume Method in Computational Fluid Dynamics: An Advanced Introduction with OpenFOAM® and Matlab*. Springer International Publishing, 2016. ISBN: 9783319168746. DOI: 10.1007/978-3-319-16874-6. URL: http://dx.doi.org/10.1007/978-3-319-16874-6 (cit. alle pp. 23, 24, 26, 27, 29–31, 33–35).

[25] F. Moukalled, L. Mangani e M. Darwish. *The Finite Volume Method in Computational Fluid Dynamics: An Advanced Introduction with OpenFOAM® and Matlab*. Springer International Publishing, 2016, pp. 103–135. ISBN: 9783319168746. DOI: 10.1007/978-3-319-16874-6. URL: http://dx.doi.org/10.1007/978-3-319-16874-6 (cit. alle pp. 23, 26).

[26] *OpenFOAM website*. URL: https://www.openfoam.com (visitato il 20/05/2024) (cit. a p. 35).

[27] Christopher Greenshields e Henry Weller. *Notes on Computational Fluid Dynamics: General Principles*. Reading, UK: CFD Direct Ltd, 2022 (cit. a p. 38).

[28] *Computational resources were provided by HPC@POLITO, a project of Academic Computing within the Department of Control and Computer Engineering at the Politecnico di Torino*. URL: `http://www.hpc.polito.it` (visitato il 25/07/2024) (cit. a p. 38).

[29] Vaclav Smilauer et al. *Yade documentation*. en. 2021. DOI: `10.5281/ZENODO.5705394`. URL: `https://zenodo.org/record/5705394` (cit. a p. 39).

[30] *OpenFOAM User Guide v2112: function objects*. URL: `https://www.openfoam.com/documentation/guides/v2112/doc/guide-function-objects.html` (visitato il 26/10/2024) (cit. a p. 44).