Master in Tecniche Informatiche Avanzate per la Progettazione e la Simulazione Numerica

Studio del Comportamento Idroelastico di Strutture Galleggianti

Ilaria Lodes

Maggio 2000

Relatori:
Giorgio Fotia, CRS4
Maurizio Landrini, INSEAN

Indice

Desidero ringraziare i miei tutors Giorgio Fotia e Maurizio Landrini per avermi costantemente aiutato durante il mio stage con la loro disponibilità, la loro competenza e, non ultima, la loro amicizia.
Un ringraziamento particolare a Giuseppina, Rosa e Angelica nuove meravigliose amiche.
A Marco per esserci sempre e comunque.
Alle tredici splendide persone che hanno vissuto con me quest'avventura.
A P.G. per ....

Capitolo 1

Introduzione

Lo scopo fondamentale degli studi di idrodinamica navale, e pertanto delle sperimentazioni e simulazioni eseguite all'INSEAN (Istituto Nazionale per Studi ed Esperienze di Architettura Navale), è *l'ottimizzazione dei veicoli marini*. Ottimizzare un veicolo marino significa garantirne la sicurezza, migliorarne il comfort e incrementarne l'operatività. A questo scopo, assumono particolare rilevanza gli studi relativi all'**interazione fluido-struttura**. Quest'ultima è una problematica complessa la cui comprensione richiede lo studio di altri problemi a loro volta complessi e di ampio interesse:

- La generazione di onde
- La generazione di vorticità
- I fenomeni di impatto
- I fenomeni idroelastici

Il progetto di tesi svolto durante lo stage conclusivo del "Master in Tecniche Informatiche Avanzate per la Progettazione e la Simulazione Numerica" si proponeva di studiare il comportamento idroelastico delle strutture galleggianti.

Analizzando il campo fluidodinamico circostante un corpo galleggiante (1.1), si può notare come esso si divida in due zone diverse dal punto di vista delle caratteristiche del moto e, pertanto, della descrizione che di tale moto si può fornire. La zona più vicina al corpo è caratterizzata dal fatto che vi insorgono rilevanti fenomeni viscosi con generazione di onde e vorticità. La zona più lontana è essenzialmente sede della propagazione di moti ondosi, che possono essere efficacemente descritti come moti a potenziale. La differenza tra la fisica di questi fenomeni suggerisce di trattare le due zone come due

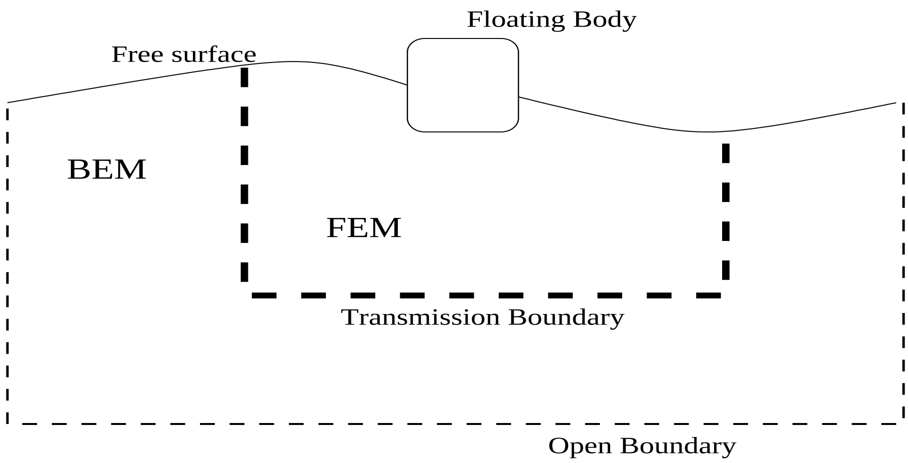

Figura 1.1: Schematizzazione del campo fluidodinamico.

diversi domini, applicando in ciascuna gli strumenti più idonei alla risoluzione del problema. Da qui, cioè, nasce l'idea di applicare algoritmi di **domain decomposition**.
Il dominio più interno, dominato da fenomeni viscosi, è meglio caratterizzato tramite le equazioni di Navier-Stokes; lo strumento computazionale più idoneo a risolvere tali equazioni è rappresentato dai **metodi agli elementi finiti**. Inoltre, gli elementi finiti sono di norma utilizzati per l'analisi strutturale e pertanto risultano adatti anche allo studio dell'interazione fluido-struttura. Per il dominio esterno, invece, risulta sufficiente la descrizione del fluido come incomprimibile ed non viscoso, dotato di moto irrotazionale. Pertanto il campo fluidodinamico risulta completamente descritto dall'equazione di conservazione della massa (che si riduce all'equazione di Laplace per il potenziale della velocità) e può essere convenientemente risolto con il **metodo agli integrali di contorno**.
Lo scopo di questo lavoro di stage consiste nell'accoppiare diversi modelli (fluido viscoso - fluido non viscoso) nelle diverse porzioni del continuo fluido-struttura e nel realizzare ed accoppiare diversi solutori (FEM - BEM) per i diversi modelli. Pertanto le fasi dello studio del campo fluidodinamico sono state pianificate nel modo seguente:

- Sviluppo di un algoritmo BEM (Boundary Element Method)
- Sviluppo di un algoritmo FEM (Finite Element Method)
- Scelta e sviluppo di un algoritmo di Domain Decomposition

- Accoppiamento

Oltre allo studio del campo fluidodinamico, sussiste la necessità di sviluppare una analisi strutturale del corpo galleggiante considerato deformabile. Purtroppo, per motivi di insufficiente disponibilità temporale, non è stato possibile occuparsi di questa parte del problema, che dunque dovrebbe essere portata avanti in uno studio successivo. Inoltre, non si è potuto affrontare il problema della risoluzione delle equazioni di Navier-Stokes: infatti, allo scopo di potersi concentrare maggiormente su alcuni aspetti, quali ad esempio gli algoritmi di accoppiamento, si è preferito semplificare il problema risolvendo anche nel dominio più interno l'equazione di Laplace. Pertanto uno degli obiettivi di una possibile prosecuzione di questo lavoro dovrebbe essere l'implementazione di un solutore per le equazioni di Navier-Stokes.
La risoluzione del problema, seppure così semplificato, ha richiesto più tempo di quanto era stato inizialmente previsto in quanto è stato necessario risolvere problemi paralleli a quello principale che, seppur necessari ad una maggiore comprensione del fenomeno, non hanno consentito di concludere il lavoro nel tempo previsto. In conclusione, sebbene siano stati sviluppati quasi tutti i tasselli necessari alla composizione del mosaico finale, non è stato possibile comporli ottenendo il codice definitivo.

Fasi del lavoro Il lavoro di stage, svolto in parte presso il CRS4 ed in parte presso l'INSEAN, è stato suddiviso in due fasi distinte. Nella prima si è concentrata l'attenzione sulla soluzione di problemi di **sloshing**, mentre nella seconda si è portato avanti il problema della **domain decomposition**.

Sloshing: Novembre 1999 – Gennaio 2000 Lo sloshing è il moto a superficie libera di un liquido all'interno di un contenitore in movimento. Si tratta di un fenomeno estremamente comune che si verifica, tra l'altro, sulle navi adibite al trasporto di petrolio o gas liquefatti e nei serbatoi di razzi o aerei. Le conseguenze dello sloshing variano dalla perdita di fluido all'incremento di sollecitazione sulle pareti dei contenitori rispetto alla condizione idrostatica, alla perdita di stabilità del veicolo. Il fenomeno potrebbe apparire lontano dal problema che ci si propone di risolvere, ossia il comportamento idroelastico dei contenitori galleggianti, se non altro perchè in quest'ultimo caso il fluido risulta esterno al contenitore mentre, nel caso dello sloshing, il liquido è interno al contenitore. Tuttavia, lo sloshing è stato scelto come caso di studio per diversi motivi:

- Come nel caso dei contenitori galleggianti, si ha a che fare con il moto a superficie libera di un fluido: questo pone dei problemi di fondamentale importanza quale quello di seguire opportunamente il movimento della superficie libera, adottando algoritmi di aggiornamento della griglia di calcolo tra un istante ed il successivo, o quello di fissare opportunamente le condizioni al contorno in corrispondenza della superficie libera stessa nonchè delle pareti del contenitore.

- Il problema dello sloshing ha consentito di studiare e di applicare degli algoritmi FEM e BEM per la risoluzione delle equazioni di Navier-Stokes e dell'equazione di Laplace, rispettivamente, in domini mobili.

L'analisi totalmente non lineare dello sloshing è stata effettuata nel dominio del tempo; si sono considerati principalmente casi di oscillazione orizzontale di un dominio bidimensionale. Tale analisi potrebbe, dunque, in futuro essere estesa prendendo in esame casi di oscillazione verticale o di rollio e introducendo la tridimensionalità del dominio.
I risultati delle simulazioni effettuate, insieme all'analisi teorica del fenomeno e dell'algoritmo FEM impiegato per la risoluzione, sono riportati in un rapporto tecnico per l'Insean ([10]). In [11] sono riassunti i risultati del confronto tra un solutore BEM e un solutore FEM. Si rimanda pertanto a [10] e [11] per lo studio dello sloshing, mentre questa relazione sarà dedicata esclusivamente alla seconda parte del lavoro.

Domain Decomposition: Febbraio 2000 – Maggio 2000 La prima fase del lavoro è stata dedicata allo studio degli algoritmi BEM per la risoluzione dell'equazione di Laplace in domini bidimensionali. È stato innanzitutto analizzato un codice per lo sloshing in un contenitore rettangolare: questo ha consentito di studiare il metodo dei punti di collocazione e il modo in cui vengono imposte le condizioni al contorno sulla superficie libera e sulle pareti del contenitore; inoltre, ha consentito di analizzare il problema dell'integrazione nel tempo allo scopo di seguire l'evoluzione della superficie libera.
Quindi si è iniziato lo sviluppo del codice BEM da utilizzarsi per la domain decomposition. Inizialmente, si è fatto riferimento ad un problema matematico semplice, per il quale fosse disponibile la soluzione analitica, allo scopo di verificare la validità del codice ed effettuare alcune prove di convergenza. I risultati di questa fase sono riassunti nel capitolo 2.
Utilizzando poi come solutore l'algoritmo BEM sviluppato, sono stati analizzati ed applicati vari algoritmi di domain decomposition suggeriti in letteratura; i test effettuati hanno portato alla scelta del *metodo Dirichlet - Neumann* per l'accoppiamento dei due solutori (capitolo 3).
Successivamente è stato sviluppato un codice agli elementi finiti lineari per

la risoluzione dell'equazione di Laplace in un dominio bidimensionale. Anche qui è stato considerato dapprima il problema semplificato con condizioni al contorno di Dirichlet su tutto il dominio (capitolo 4). In questo modo si è evidenziato il problema del calcolo accurato delle derivate del potenziale sul contorno. Dopo svariati tentativi, si è dimostrato euristicamente come il metodo della *Patch Recovery* suggerito da Zienkiewicz e Zhu ([18],[19]) fornisca risultati estremamente soddisfacenti (capitolo 5). Analogamente risultati soddisfacenti si ottengono considerando condizioni al contorno Dirichlet e Neumann su diverse porzioni della frontiera.
Per verificare, quindi, che il livello di approssimazione ottenuto sia sufficiente a consentire una adeguata rappresentazione dell'evoluzione della superficie libera, si è realizzato un codice per l'analisi dell'evoluzione di una *standing wave* in un contenitore rettangolare. Questo ha permesso anche di implementare un algoritmo per l'integrazione nel tempo e di puntualizzare il modo in cui devono essere imposte le condizioni al contorno, in particolare in corrispondenza dei punti di contatto (capitolo 6).
Si può dunque concludere che i passi successivi, da compiere in un eventuale prosecuzione del lavoro, consistono nel realizzare l'effettivo accopiamento dei due codici, nell'introdurre il corpo galleggiante (eventualmente considerato prima rigido e poi flessibile) e, infine, nel sostituire l'equazione di Laplace con le equazioni di Navier-Stokes nel dominio più interno.

Capitolo 2

Algoritmo BEM

Introduzione L'analisi e lo sviluppo del codice BEM, destinato all'applicazione nel codice di *Domain Decomposition*, si sono svolti attraverso due fasi:

- Analisi di un codice BEM per lo sloshing in contenitori rettangolari
- Sviluppo di un codice BEM per il problema di Domain Decomposition

La prima fase è servita essenzialmente ad acquisire familiarità con le caratteristiche dei metodi agli integrali di contorno, con le equazioni e le condizioni al contorno caratteristiche dei problemi a superficie libera, e infine con alcuni algoritmi per l'integrazione nel tempo. Nella seconda fase, si è applicato il metodo BEM alla risoluzione di un problema matematico opportunamente scelto.

Un codice BEM per lo sloshing in contenitori rettangolari Il codice analizzato ha una struttura che si articola in tre fasi:

1. per un dato istante, noti i dati al contorno, si risolve l'equazione di Laplace e si ricava il potenziale e la sua derivata normale in ogni punto del contorno (e virtualmente in ogni punto del dominio);
2. si calcolano le forze sulle pareti del contenitore;
3. si applicano le condizioni dinamiche per l'evoluzione della superficie libera e si incrementa la variabile tempo.

Tale struttura è stata poi estesa al codice successivamente sviluppato, ad eccezione del calcolo delle forze che, al momento, non è stato preso in considerazione. Nel seguito sono riassunte le caratteristiche salienti del codice per lo sloshing e, in particolare, quelle che sono risultate utili alla stesura del codice finale.

Soluzione dell'equazione di Laplace Nell'ipotesi di incomprimibilità di un liquido e di moto irrotazionale, l'equazione di continuità, che sotto tali ipotesi descrive completamente il campo fluidodinamico in un certo istante, può essere scritta in funzione del potenziale di velocità φ, ottenendo l'equazione di Laplace:

$$\nabla^2\varphi = 0 \tag{2.1}$$

Se si introduce la soluzione fondamentale dell'equazione di Laplace, che in 2D vale

$$G = \ln|P - Q| \tag{2.2}$$

e si applicano opportunamente le formule di Green, si ricava l'equazione integrale

$$\int_{\partial\Omega} \varphi\frac{\partial G}{\partial n} - G\frac{\partial\varphi}{\partial n}dS = \begin{cases} 0 & P \in R^2/\Omega \\ \pi\varphi(P) & P \in \partial\Omega \\ 2\pi\varphi(P) & P \in \mathring{\Omega} \end{cases} \tag{2.3}$$

Tramite questa equazione, se si conosce il valore del potenziale su $\partial\Omega$, è possibile ricavare il potenziale in qualunque punto del dominio. Tuttavia, nella maggior parte dei casi di pratico interesse, il potenziale è noto solo su una parte della frontiera (nel nostro caso sulla superficie libera) e pertanto è necessario innanzitutto determinare il potenziale su tutti i punti della frontiera (ovvero sulle pareti del contenitore).
Considerato un punto P sul contorno, possono verificarsi due eventualità:

- Il punto P si trova sulla porzione della frontiera con condizioni al contorno di tipo Dirichlet: il potenziale in P è noto, mentre è incognita la derivata normale del potenziale; l'equazione 2.3 diventa:

$$-\int_D \frac{\partial\varphi}{\partial n}GdS + \int_N \varphi\frac{\partial G}{\partial n}dS = \alpha\varphi(P) - \int_D \varphi\frac{\partial G}{\partial n}dS + \int_N \frac{\partial\varphi}{\partial n}GdS \tag{2.4}$$

- Il punto P si trova sulla porzione della frontiera con condizioni al contorno di tipo Neumann: il potenziale in P è incognito, mentre è nota la derivata del potenziale; l'equazione 2.3 diventa:

$$-\int_D \frac{\partial\varphi}{\partial n}GdS - \alpha\varphi(P) + \int_N \varphi\frac{\partial G}{\partial n}dS = -\int_D \varphi\frac{\partial G}{\partial n}dS + \int_N \frac{\partial\varphi}{\partial n}GdS \tag{2.5}$$

L'algoritmo per la risoluzione numerica dell'equazione di Laplace prevede innanzitutto la discretizzazione della frontiera: indichiamo con N_D il numero di pannelli sulla frontiera Dirichlet e con N_N il numero di pannelli sulla frontiera Neumann. Il punto P viene considerato di volta in volta coincidente con uno degli estremi di tali pannelli. In questo modo, si avranno $N_D + 1$ incognite $\partial\varphi/\partial n$, relative ai nodi sulla frontiera Dirichlet, e $N_N + 1$ incognite φ, relative ai nodi sulla frontiera Neumann; corrispondentemente si possono scrivere $N_D + 1 + N_N + 1$ equazioni.
Nel codice in esame, i pannelli utilizzati sono lineari (segmenti che approssimano il contorno) ed ugualmente lineari sono le funzioni di forma. Se si considera, ad esempio, il primo integrale a primo membro dell'equazione 2.4, la discretizzazione produce:

$$\int_D \frac{\partial\varphi}{\partial n} G dS = \sum_{j=1}^{N_D} \int_{\xi_j}^{\xi_{j+1}} \frac{\partial\varphi}{\partial n}(\xi) G(P, Q(\xi)) d\xi \tag{2.6}$$

Ponendo $\partial\varphi/\partial n = \psi$ ed introducendo le funzioni di forma si ricava:

$$\frac{\psi - \psi_j}{\psi_{j+1} - \psi_j} = \frac{\xi - \xi_j}{\xi_{j+1} - \xi_j} \tag{2.7}$$

ovvero

$$\psi = \psi_{j+1} \frac{\xi - \xi_j}{\Delta\xi} + \psi_j \frac{\xi_{j+1} - \xi}{\Delta\xi} \tag{2.8}$$

L'integrale diventa

$$\begin{aligned} \int_D \frac{\partial\varphi}{\partial n} G dS &= \sum_{j=1}^{N_D} \left[\int_{\xi_j}^{\xi_{j+1}} \psi_{j+1} \frac{\xi - \xi_j}{\Delta\xi} G + \psi_j \frac{\xi_{j+1} - \xi}{\Delta\xi} G \right] \\ &= \sum_{j=1}^{N_D} \left[\psi_{j+1} \int_{\xi_j}^{\xi_{j+1}} \frac{\xi - \xi_j}{\Delta\xi} G + \psi_j \int_{\xi_j}^{\xi_{j+1}} \frac{\xi_{j+1} - \xi}{\Delta\xi} G \right] \\ &= \sum_{j=1}^{N_D} \psi_{j+1} A(i, j+1) + \psi_j A(i, j) \end{aligned} \tag{2.9}$$

dove $i = 1, \ldots, N_D + 1$.
Procedendo analogamente per entrambe le equazioni, si perverrà ad un sistema lineare del tipo:

$$A * X = B * Y \tag{2.10}$$

la cui dimensione è $N_{sys} = N_D + 1 + N_N + 1$. Il vettore X contiene le incognite: prima le $N_D + 1$ incognite $\frac{\partial\varphi}{\partial n}$ e poi le $N_N + 1$ incognite φ. Il vettore

Y contiene i valori noti di φ nelle prime $N_D + 1$ posizioni ed i valori noti di $\frac{\partial\varphi}{\partial n}$ nelle restanti $N_N + 1$ posizioni.
Per ripristinare la continuità (poiché il primo e l'ultimo punto della superficie libera coincidono rispettivamente con l'ultimo ed il primo punto della parete del contenitore), si devono imporre le seguenti condizioni:

- $X(N_D + 1 + 1)$ e $X(N_{sys})$ = al valore noto del potenziale;
- le righe $N_D + 1 + 1$ e N_{sys} delle matrici A e B = 0;
- $A(N_D + 1 + 1, N_D + 1 + 1) = 1$
- $A(N_{sys}, N_{sys}) = 1$

ovvero, in corrispondenza dei punti della parete che appartengono anche alla superficie libera il potenziale è noto e l'equazione corrispondente si riduce ad una identità.

Risolto tale sistema, si avranno φ e $\partial\varphi/\partial n$ su tutta la frontiera e quindi sarà possibile ricavare φ e $\partial\varphi/\partial n$ in tutto il dominio.

Differenziando il potenziale sulla superficie libera, si ricava la velocità sulla superficie libera stessa (in particolare le due componenti lungo x e lungo y); per quanto riguarda i punti di contatto, però, si deve imporre una condizione di compatibilità: la componente della velocità normale alla parete del contenitore, calcolata considerando il punto come appartenente alla superficie libera, deve essere uguale alla $\partial\varphi/\partial n$ imposta in corrispondenza della parete del contenitore:

$$\left.\frac{\partial\varphi}{\partial n}\right|_{Bo} = \nabla\varphi|_{Fs} \cdot \mathbf{n}_{Bo} \tag{2.11}$$

Note le velocità in corrispondenza della superficie libera, si ricava immediatamente l'evoluzione della superficie libera stessa integrando nel tempo le condizioni di superficie libera.

L'avanzamento nel tempo Nel codice in esame, l'avanzamento nel tempo è realizzato attraverso l'integrazione delle condizioni di superficie libera mediante un algoritmo che può essere scelto tra:

1. Eulero esplicito
2. Runge-Kutta del secondo ordine
3. Runge-Kutta del quarto ordine

Le condizioni di superficie libera sono state scritte in due modi alternativi.

- **Forma Lagrangiana** Si assume che i punti di griglia coincidano con particelle sulla superficie libera e quindi si muovano con la velocità di tali particelle. Le equazioni che devono essere integrate sono le seguenti:

$$\frac{D\varphi}{Dt} = \frac{1}{2}|\nabla\varphi|^2 - g\eta \tag{2.12}$$
$$\frac{D\mathbf{P}}{Dt} = \nabla\varphi \tag{2.13}$$

  dove $\mathbf{P}$ è il vettore di posizione di un punto della superficie libera.

- **Forma Euleriana** Si assume che i punti di griglia sulla superficie libera si muovano con velocità $\mathbf{w} = [0, \dot{\eta}]$; $\dot{\eta}$, variazione temporale dell'elevazione d'onda, è data da

$$\dot{\eta} = -\frac{\partial\varphi}{\partial x}\frac{\partial\eta}{\partial x} + \frac{\partial\varphi}{\partial y} \tag{2.14}$$

  In tal caso le equazioni di evoluzione diventano:

$$\begin{aligned}\frac{\delta\varphi}{\delta t} &= \mathbf{w}\cdot\nabla\varphi - \frac{1}{2}|\nabla\varphi|^2 - g\eta \\ &= \dot{\eta}\frac{\partial\varphi}{\partial y} - \frac{1}{2}|\nabla\varphi|^2 - g\eta \qquad (2.15)\\ \frac{\delta\mathbf{P}}{\delta t} &= \mathbf{w} \qquad (2.16)\end{aligned}$$

  dove

$$\dot{\eta} = -\frac{\partial\varphi}{\partial x}\frac{\partial\eta}{\partial x} + \frac{\partial\varphi}{\partial y} \tag{2.17}$$

In ogni caso, l'avanzamento nel tempo consiste nell'integrazione di equazioni differenziali che possono essere così schematizzate:

$$\frac{dy}{dt} = f(\mathbf{x}, t) \tag{2.18}$$

dove y rappresenta una generica funzione, mentre $\mathbf{x}$ rappresenta la configurazione assunta dal dominio all'istante t.
I tre schemi di integrazione adottabili possono essere sintetizzati nel modo seguente:
per $i = 1, \ldots, ikutta$ con $ikutta = 1, 2, 4$ rispettivamente per i casi Eulero esplicito, Runge-Kutta secondo e quarto ordine, si pone:

1. $t_i = t_0 + \beta_k \Delta t$

2. $\mathbf{x}_i = \mathbf{x}_0 + \beta_k f(t_{i-1}, \mathbf{x}_{i-1}) \Delta t$

e quindi si ricava y

$$y = y_0 + \left[\sum_{i=1}^{ikutta} \alpha_k f(t_i, \mathbf{x}_i) \right] \Delta t \tag{2.19}$$

I valori dei coefficienti α_k e β_k sono riportati in tabella 2.1.

Metodo	i	α	β
Eulero esplicito	1	1	0
Runge-Kutta II	1	0.5	0
	2	0.5	1
Runge-Kutta IV	1	1/6	0
	2	1/3	0.5
	3	1/3	0.5
	4	1/6	1

Tabella 2.1: Coefficienti per il metodo Runge-Kutta.

Risultati Sono state eseguite alcune simulazioni di prova e i risultati ottenuti sono stati confrontati con quelli provenienti da un altro codice BEM che impiega come funzioni di forma delle B-spline e che utilizza un algoritmo di avanzamento in tempo *Runge-Kutta del quarto ordine*.
Sono stati considerati diversi casi di *sway*; in particolare la Fig. 2.1 riporta l'andamento dell'elevazione d'onda in un punto di contatto per il caso di oscillazione orizzontale di un contenitore rettangolare largo 8m e alto 1m; l'ampiezza dell'oscillazione è di 0.0372m e la pulsazione dell'oscillazione è 1.19954rad/s, mentre la prima frequenza propria è 1.19958rad/s: sono confrontati i risultati ottenuti da G. Colicchio in [5] e quelli ottenuti con il codice sopra descritto sia utilizzando la condizione dinamica lagrangiana sia utilizzando la condizione euleriana. In ogni caso la soluzione ottenuta in [5] presenta altezze d'onda più elevate e oscillazioni maggiori. Inoltre se si utilizza la condizione dinamica lagrangiana è necessario utilizzare un maggior numero di pannelli per riuscire a seguire la deformazione della superficie libera, e comunque le due soluzioni (*lagrangiana* ed *euleriana*) risultano sostanzialmente coincidenti. Anche la forma della superficie libera ottenuta nei due casi per diversi istanti di tempo è sostanzialmente identica.

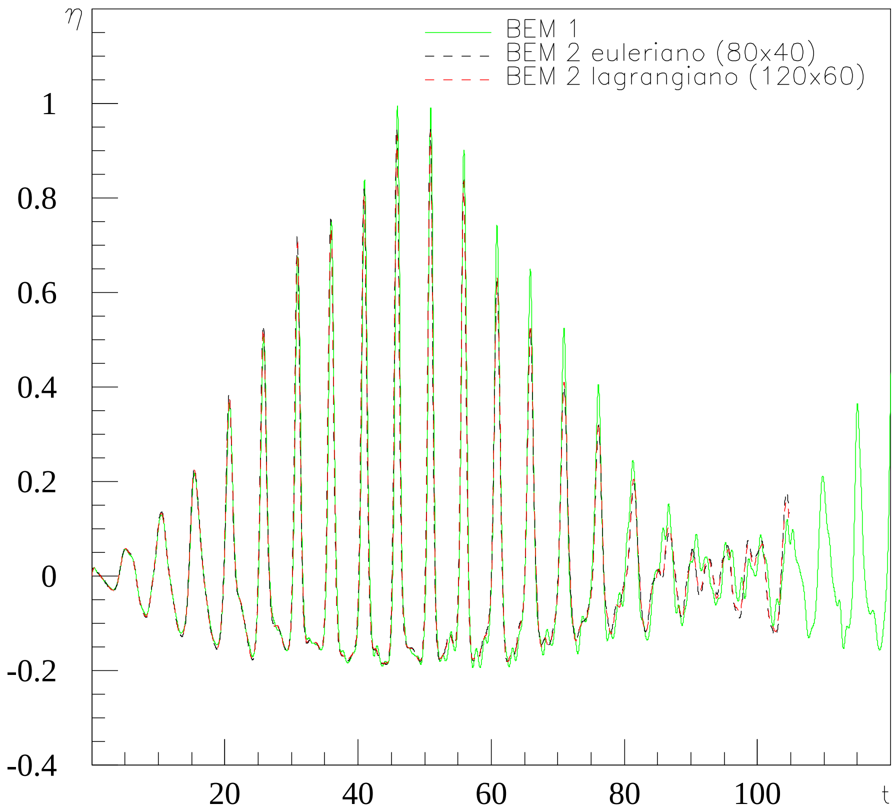

Figura 2.1: Andamento temporale dell'elevazione d'onda in un punto di contatto: confronto tra i risultati di [5] e quelli ottenuti con il codice qui descritto con condizione dinamica di superficie libera lagrangiana ed euleriana. Caso di dominio rettangolare 8×1, $A = 0.0372$, $\omega = 1.19954$.

Algoritmo BEM per il problema della Domain Decomposition Il codice BEM sviluppato risolve l'equazione di Laplace su un dominio limitato Ω, sulla cui frontiera $\partial\Omega$ possono essere applicate condizioni al contorno di Dirichlet e/o di Neumann. In un primo tempo, è stato considerato un caso semplice, un problema matematico, per il quale sia disponibile la soluzione analitica onde poter validare il codice ed effettuare alcune prove di convergenza. Il dominio preso in esame è quello racchiuso tra due circonferenze concentriche immerse in un piano xy; in esso è definita la funzione

$$\varphi = e^{ky} \sin kx \tag{2.20}$$

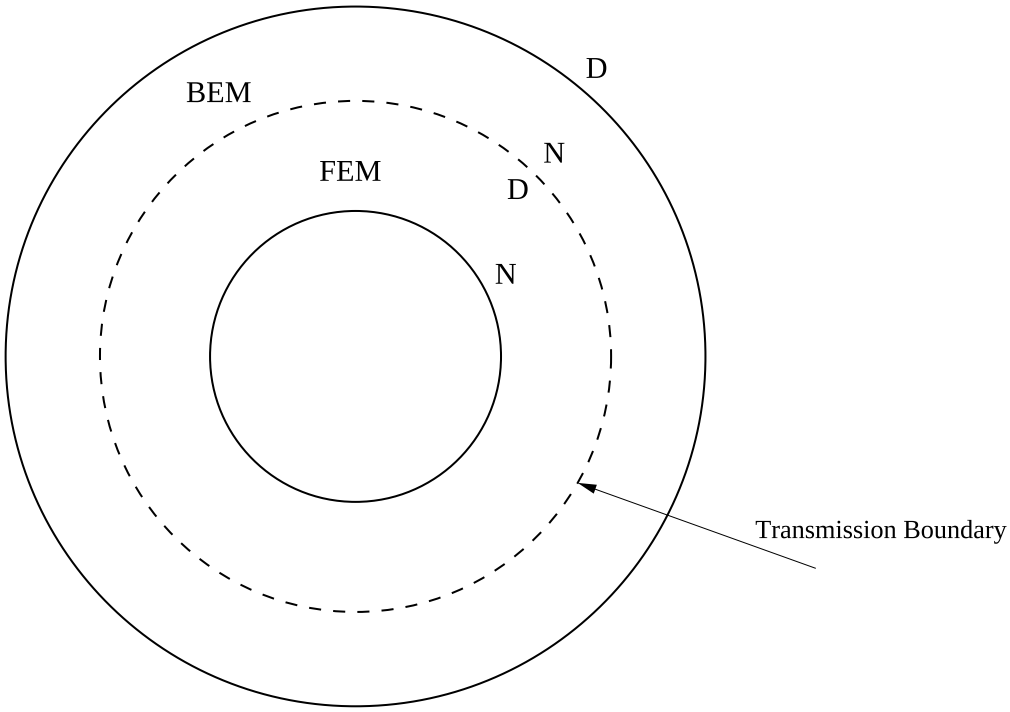

Figura 2.2: Schematizzazione del dominio Ω e delle condizioni al contorno applicate.

Sulla frontiera di tale dominio vengono imposte le seguenti condizioni al contorno:

- sulla circonferenza esterna è assegnato il valore di φ (Condizione di Dirichlet);

- sulla circonferenza interna è assegnato il valore di $\partial\varphi/\partial n$ (Condizione di Neumann).

In figura 2, è evidenziato il fatto che il dominio verrà successivamente suddiviso in due parti: quella più esterna in cui si applicherà il metodo BEM e quella più interna in cui si applicherà il metodo FEM. In questa fase del lavoro, però, su tutto il dominio viene applicato il metodo BEM e la *transmission boundary* non ha alcun significato.
Su tale dominio e con le condizioni al contorno sopra descritte si vuole risolvere l'equazione di Laplace (problema stazionario). Il sistema dato dall'equazione di Laplace e dalle condizioni al contorno è stato risolto con il metodo BEM descritto precedentemente; l'unica differenza nell'algoritmo è data dal fatto che le due porzioni di frontiera (Dirichlet e Neumann) si richiudono su se stesse e pertanto il primo e l'ultimo punto di ciascuna coincidono. Quindi, se si discretizza la frontiera Dirichlet con N_D pannelli e la frontiera Neumann con N_N pannelli, si avranno N_D+1 incognite $\partial\varphi/\partial n$ e N_N+1 incognite φ ($N_{sys} = N_D+1+N_N+1$). Per l'ultimo punto e il primo di ciascuna porzione di frontiera è necessario ripristinare la continuità ponendo uguali a zero la $X(N_D+1)$ e $X(N_{sys})$, e così pure le righe N_D+1 e N_{sys} della matrice A ad eccezione di:

- $A(N_D+1, N_D+1) = 1$
- $A(N_D+1, 1) = -1$
- $A(N_{sys}, N_{sys}) = 1$
- $A(N_{sys}, N_D+1+1) = -1$

Risolto il sistema discretizzato, si avranno φ e $\partial\varphi/\partial n$ su tutta la frontiera e quindi sarà possibile ricavare φ e $\partial\varphi/\partial n$ in tutto il dominio.

Su questo problema prototipo, sono quindi state studiate euristicamente le proprietà di convergenza dell'algoritmo: in particolare si è valutato l'andamento dell'errore su φ e su $\partial\varphi/\partial n$, sia come valore massimo che in norma-2, al variare della dimensione dei pannelli utilizzatii. La dimensione dei pannelli è rappresentata dal rapporto $2\pi R/N$, dove il numero dei pannelli N è scelto in modo tale che tale rapporto risulti uguale per le due circonferenze interna ed esterna. L'errore commesso su φ è rappresentato in figura 2.3; nella stessa figura sono riportati anche gli errori normalizzati rapportandoli rispettivamente al massimo e alla norma-2 della soluzione analitica.
La figura 2.4 riporta analogamente l'errore massimo e la norma-2 dell'errore nella valutazione di $\partial\varphi/\partial n$.
L'errore, che diminuisce all'infittirsi della discretizzazione, è generalmente

contenuto. Se si considerano gli errori non normalizzati, l'errore su $\partial\varphi/\partial n$, valutato sulla circonferenza più esterna, è maggiore in quanto la y sulla circonferenza esterna è maggiore e ciò incide marcatamente sull'esponenziale; gli errori normalizzati, invece, sono tutti dello stesso ordine di grandezza.

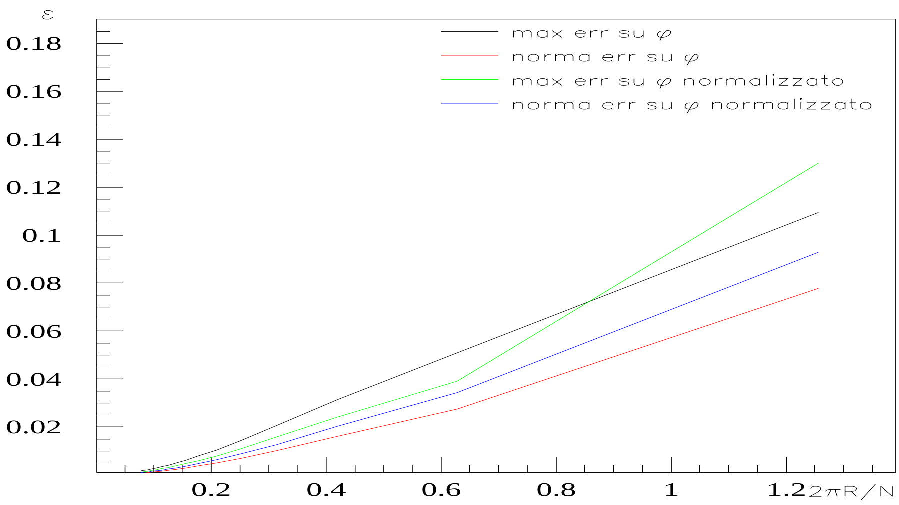

Figura 2.3: Errore massimo e norma-2 dell'errore commesso sulla valutazione di φ.

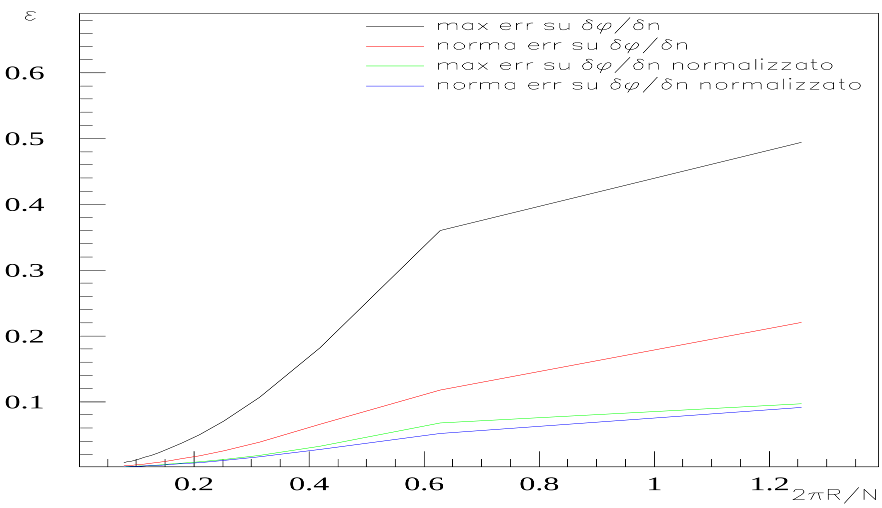

Figura 2.4: Errore massimo e norma-2 dell'errore commesso sulla valutazione di $\partial\varphi/\partial n$.

Capitolo 3

Domain Decomposition

Introduzione Gli algoritmi di *Domain Decomposition* vengono generalmente impiegati per ridurre lo sforzo computazionale nel caso di problemi a grande scala, in particolare per problemi caratterizzati da equazioni di campo ellittiche. In letteratura, sono stati proposti diversi metodi di domain decomposition, alla base dei quali c'è il principio di utilizzare opportuni sottodomini in modo tale che la soluzione nell'insieme dei sottodomini corrisponda alla soluzione ricercata nel dominio originario. L'efficienza del metodo dipende dal tipo di problema e dal metodo numerico applicato. Come evidenziato in [8], l'applicazione di tali metodi allo studio della della propagazione di onde non lineari con metodi BEM presenta numerosi vantaggi, tra i quali la riduzione dello sforzo computazionale e la possibilità di ricorrere al calcolo parallelo.
Nel caso specifico del problema in esame, l'introduzione di un algoritmo di domain decomposition presenta anche il vantaggio di poter sfruttare le caratteristiche positive degli algoritmi BEM e FEM nelle diverse porzioni del dominio in cui si fa ricorso a diversi modelli.
I metodi che sono stati presi in considerazione (detti *Iterative Substructuring Methods*) consistono nella suddivisione del dominio computazionale in due sottodomini e nella realizzazione di una procedura iterativa di accoppiamento che genera una serie di condizioni al contorno all'interfaccia tra i due sottodomini. Questa serie dovrebbe convergere alle condizioni che corrispondono alla soluzione del problema originario. Pertanto la soluzione del problema decomposto dovrebbe convergere alla soluzione del problema originario.
Nel seguito, vengono illustrati i metodi che sono stati analizzati ed implementati, con particolare riguardo al metodo *Dirichlet-Neumann* che è stato poi scelto come metodo ottimale.

Formulazione del problema e Metodo Dirichlet-Neumann Inzialmente si è studiata la formulazione del problema, come presentata in [14], facendo riferimento all'equazione di Laplace con dati al contorno di Dirichlet non omogenei:

$$\begin{cases} \nabla^2\varphi &= 0 & \text{su } \Omega \\ \varphi &= f & \text{su } \partial\Omega \end{cases} \tag{3.1}$$

Se il dominio Ω viene diviso in due sottodomini Ω_1 e Ω_2 e si indica con Γ la frontiera di trasmissione, la forma *Multi-Domain* corrispondente è la seguente:

$$\begin{cases} \nabla^2\varphi_1 &= 0 & \text{in } \Omega_1 \\ \varphi_1 &= f & \text{su } \partial\Omega_1 \cap \partial\Omega \\ \varphi_1 &= \varphi_2 & \text{su } \Gamma \\ \frac{\partial\varphi_2}{\partial n} &= \frac{\partial\varphi_1}{\partial n} & \text{su } \Gamma \\ \varphi_2 &= f & \text{su } \partial\Omega_2 \cap \partial\Omega \\ \nabla^2\varphi_2 &= 0 & \text{in } \Omega_2 \end{cases} \tag{3.2}$$

La difficoltà nella risoluzione di questo problema consiste nel fatto che non sono in generale noti i valori del potenziale e delle sue derivate sulla frontiera di trasmissione. Pertanto si pone il problema di imporre adeguatamente le condizioni al contorno in corrispondenza di tale frontiera e quindi di ricorrere a metodi iterativi che a partire da stime iniziali φ_1^0 e φ_2^0, generino una successione di soluzioni $\{\varphi_1^k\}$ e $\{\varphi_2^k\}$ che, si spera, convergano a φ_1 e φ_2 (da qui il nome di *Iterative Substructuring Methods*).
Il metodo adottato è quello definito *Metodo Dirichlet-Neumann*: su un lato della frontiera di trasmissione vengono imposte condizioni di tipo Dirichlet, mentre sull'altro vengono imposte condizioni di tipo Neumann. In questo modo, per ciascuna iterazione, da un lato si ricava la $\partial\varphi/\partial n$ che viene poi utilizzata come condizione al contorno per il secondo sottodominio, dall'altro si ricava la φ che viene quindi trasmessa al primo sottodominio.
Nel caso specifico con dati al contorno di Dirichlet sulla frontiera esterna e di Neumann su quella interna (vedi Fig. 2, il problema è stato scritto nel modo seguente: dati $\lambda^0 = 0$ e $\frac{\partial\varphi_2^0}{\partial n} = 0$ (dove λ^0 rappresenta il valore iniziale di φ sulla frontiera di trasmssione), si risolvono per ogni $k \geq 0$ i sistemi:

$$\begin{cases} \nabla^2\varphi_1^{k+1} &= 0 & \text{in} & \Omega_1 \\ \varphi_1^{k+1} &= \hat{\varphi}_1 & \text{su} & \mathcal{D}_1 \\ \frac{\partial\varphi_1^{k+1}}{\partial n_1} &= -\frac{\partial\varphi_2^k}{\partial n_2} & \text{su} & \mathcal{N}_1 \end{cases} \tag{3.3}$$

$$
\begin{cases}
\nabla^2 \varphi_2^{k+1} & = & 0 & \text{in} & \Omega_2 \\
\varphi_2^{k+1} & = & \lambda^{k+1} & \text{su} & \mathcal{D}_2 \\
\frac{\partial \varphi_2^{k+1}}{\partial n_2} & = & \frac{\partial \hat{\varphi}}{\partial n} & \text{su} & \mathcal{N}_2
\end{cases}
\tag{3.4}
$$

dove

$$
\lambda^{k+1} = \theta \varphi_1^{k+1} + (1-\theta)\lambda^k \tag{3.5}
$$

dove θ è un parametro di rilassamento, necessario a garantire la convergenza. Come sottolineato in [14], lo stesso metodo senza rilassamento (cioè con $\theta = 1$) non converge necessariamente a meno che non vengano fatte particolari assunzioni sulla geometria dei sottodomini. Si è quindi ricercato euristicamente il valore di θ che garantisse una più rapida convergenza: dalla Fig. 3.1 appare evidente come tutti i valori considerati garantiscano una rapida convergenza e già dopo tre iterazioni il residuo risulti sufficientemente piccolo; il valore $\theta = 0.7$ sembra comunque garantire i risultati migliori. Chiaramente tale risultato resta momentaneamente limitato al caso particolare preso in considerazione.

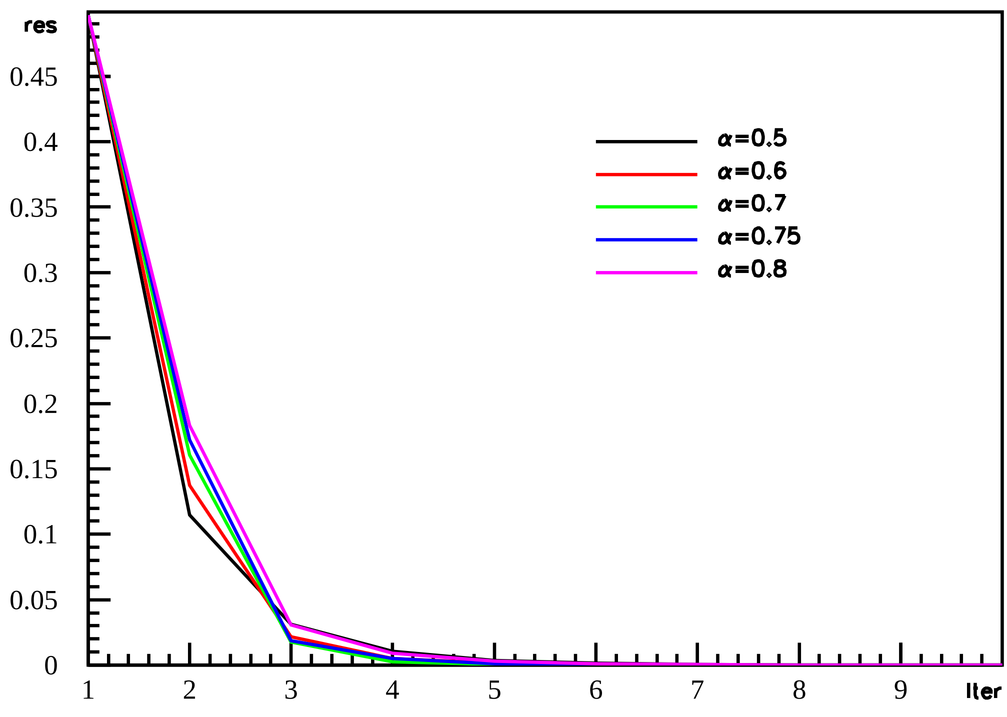

Figura 3.1: Ricerca euristica del valore ottimale di θ: metodo Dirichlet-Neumann.

Facendo riferimento al problema prototipo e utilizzando il metodo appena descritto, si è analizzato l'errore commesso al variare della dimensione della discretizzazione spaziale utilizzata. La Fig. 3.2 rappresenta in scala logaritmica, per il caso di griglia conforme, l'errore (massimo e in norma-2) sia su φ (valutato sulla frontiera interna) che su $\frac{\partial \varphi}{\partial n}$ (valutato sulla frontiera esterna), confrontato con quello che si ottiene in assenza di Domain Decomposition. Si può notare come gli andamenti delle curve siano del tutto analoghi, e addirittura gli errori su $\frac{\partial \varphi}{\partial n}$ sembrano diminuire più rapidamente che nel caso ad unico dominio.
È stato considerato anche il caso di griglia non conforme, ovvero il caso in cui sulla frontiera di trasmissione i nodi dei due domini non coincidono: nel caso in cui in entrambi i domini si impieghi un algoritmo BEM, la non conformità non rappresenta una necessità. Tuttavia, nel caso in cui si adotti per uno dei domini un algoritmo FEM e che i grigliatori impiegati siano del tutto indipendenti, si capisce come la non conformità sia un'eventualità tutt'altro che rara. Per questo motivo, si è voluto prendere in considerazione questa possibilità sin da questa fase.
Nel caso di griglie non conformi, si presenta il problema di valutare, in base ai dati calcolati in corrispondenza dei nodi di una griglia, i dati in corrispondenza dei nodi dell'altra griglia. Il problema è stato risolto costruendo, in corrispondenza della frontiera di trasmissione, una spline utilizzando i valori noti in corrispondenza dei nodi della prima discretizzazione e quindi interpolando per ottenere i dati in corrispondenza dei nodi della seconda discretizzazione. Tale procedimento ha fornito dei risultati estremamente soddisfacenti. La Fig. 3.3 mostra come l'andamento del diagramma sia simile a quello illustrato precedentemente, salvo il fatto che, laddove la griglia venga maggiormente infittita da un lato, l'errore corrispondente decresce più rapidamente.

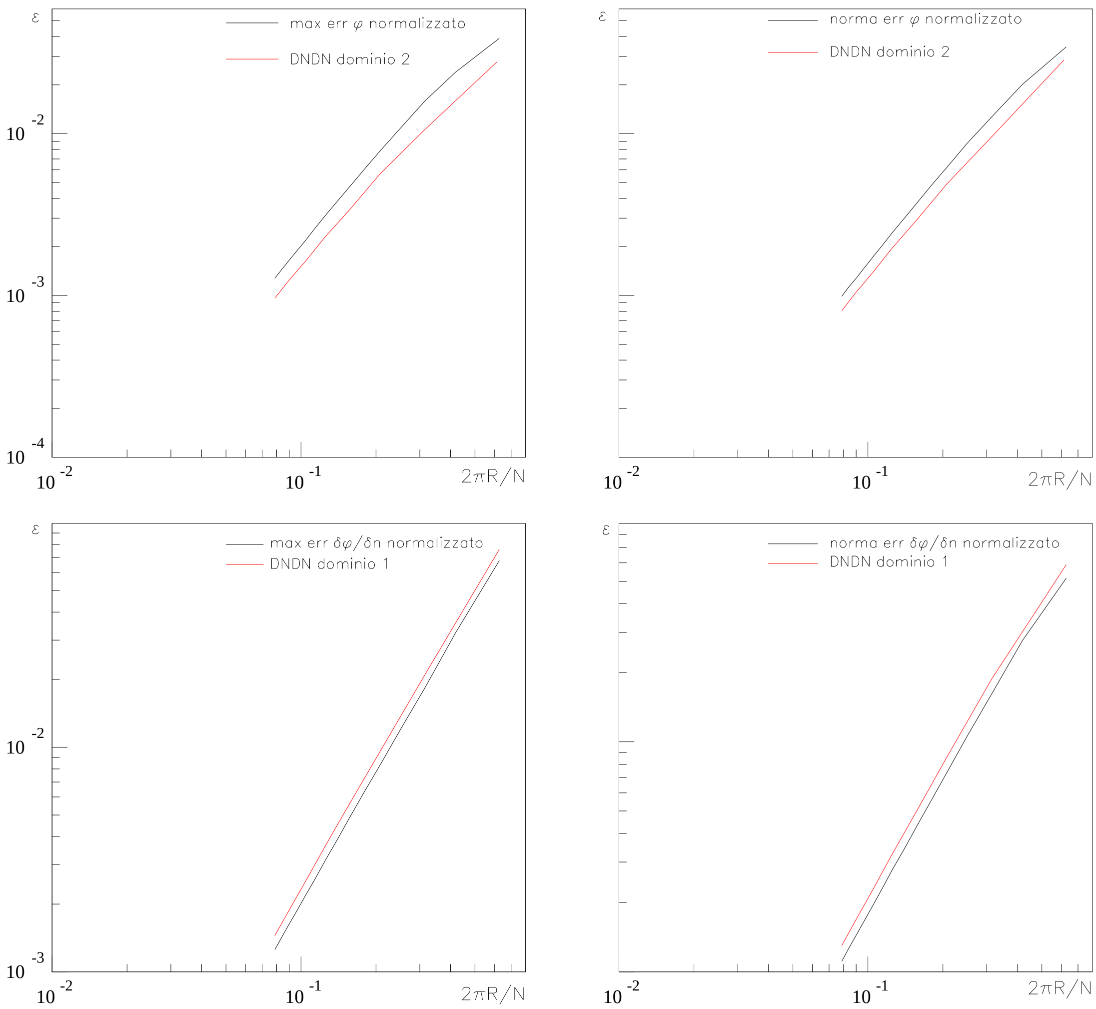

Figura 3.2: Errore al variare della dimensione dei pannelli utilizzati: caso della griglia conforme.

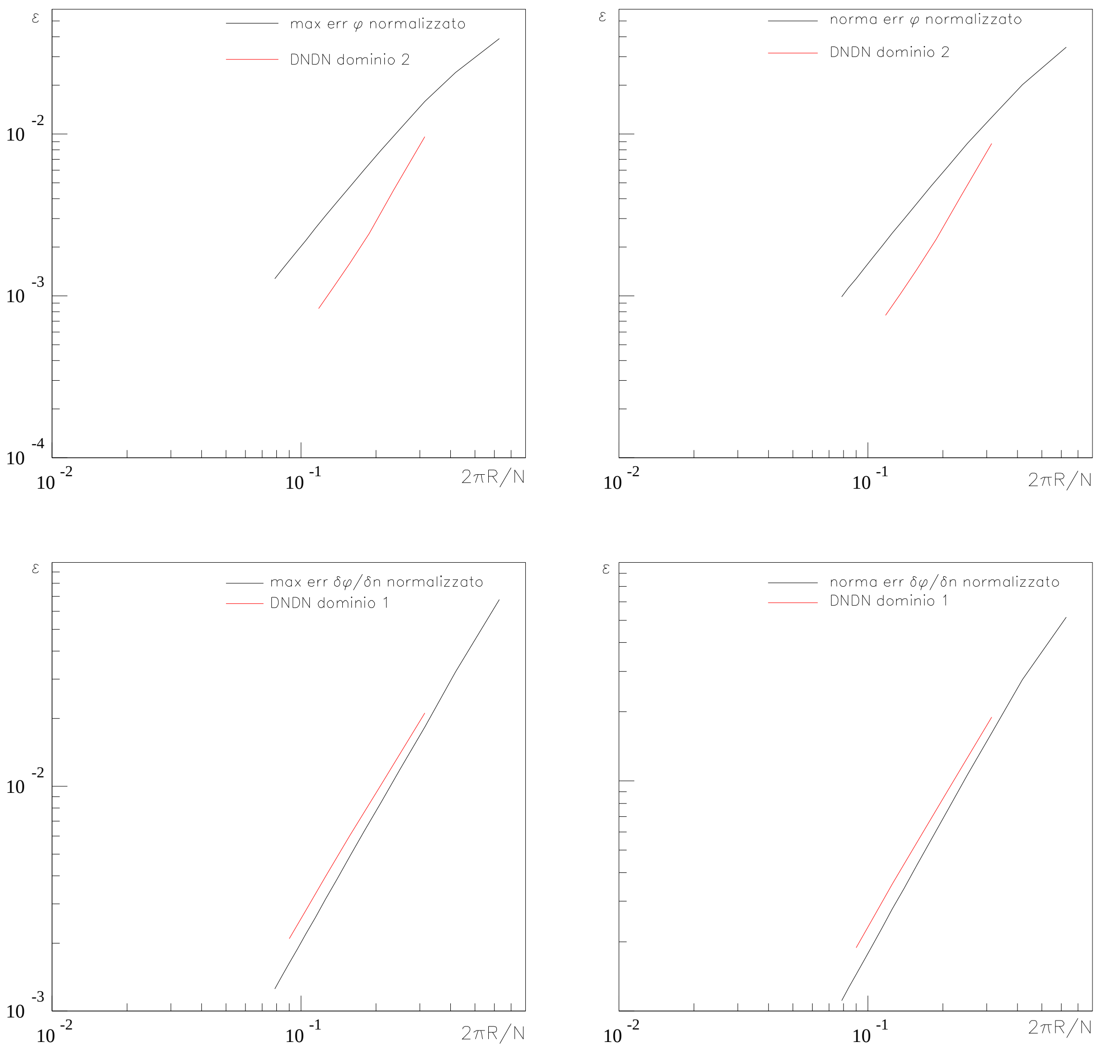

Figura 3.3: Errore al variare della dimensione dei pannelli utilizzati: caso della griglia non conforme.

Un altro metodo per la domain decomposition Oltre al metodo Dirichlet-Neumann appena descritto, è stato preso in considerazione ed implementato un secondo metodo iterativo, che definiremo "misto". Il metodo si articola in due passi:

1. Condizione al contorno di Neumann su entrambi i lati della frontiera di trasmissione: i due valori di φ così ricavati vengono mediati e il valore φ_m viene utilizzato come dato al contorno nel passo 2;

2. Condizione al contorno di Dirichlet su entrambi i lati della frontiera di trasmissione: i due valori di $\partial\varphi/\partial n$ ricavati vengono mediati e $\partial\varphi/\partial n_m$ viene utilizzato come dato al contorno nel passo 1 dell'iterazione successiva.

Se, però, si fosse considerato il problema prima descritto, con condizioni al contorno di Neumann sulla frontiera interna, nel passo 1 si sarebbe dovuto risolvere un problema con condizioni di Neumann su tutta la frontiera, problema che risulta indeterminato. Pertanto il problema originario è stato modificato considerando condizioni di Dirichlet sia sulla frontiera interna che su quella esterna.
Il problema risolto può dunque essere schematizzato nel modo seguente:

- Passo 1

$$\begin{cases} \nabla^2\varphi_1^{k+1} = 0 & \text{in} \quad \Omega_1 \\ \varphi_1^{k+1} = \hat{\varphi}_1 & \text{su} \quad \mathcal{D}_1 \\ \frac{\partial\varphi_1}{\partial n_1}^{k+1} = -\frac{\partial\varphi}{\partial n}_m^k & \text{su} \quad \mathcal{N}_{1t} \end{cases} \tag{3.6}$$

$$\begin{cases} \nabla^2\varphi_2^{k+1} = 0 & \text{in} \quad \Omega_2 \\ \varphi_2^{k+1} = \hat{\varphi}_2 & \text{su} \quad \mathcal{D}_2 \\ \frac{\partial\varphi_2}{\partial n_2}^{k+1} = \frac{\partial\varphi}{\partial n}_m^k & \text{su} \quad \mathcal{N}_{2t} \end{cases} \tag{3.7}$$

- Passo 2

$$\begin{cases} \nabla^2\varphi_1^{k+1} = 0 & \text{in} \quad \Omega_1 \\ \varphi_1^{k+1} = \hat{\varphi}_1 & \text{su} \quad \mathcal{D}_1 \\ \varphi_1^{k+1} = \varphi_m^k & \text{su} \quad \mathcal{D}_{1t} \end{cases} \tag{3.8}$$

$$\begin{cases} \nabla^2 \varphi_2^{k+1} = 0 & \text{in} \quad \Omega_2 \\ \varphi_2^{k+1} = \hat{\varphi}_2 & \text{su} \quad \mathcal{D}_2 \\ \varphi_2^{k+1} = \varphi_m^k & \text{su} \quad \mathcal{D}_{2t} \end{cases} \tag{3.9}$$

Si noti che sia nel calcolo del φ_m che nel calcolo di $\frac{\partial \varphi}{\partial n}_m$ viene introdotto un parametro di rilassamento α; pertanto:

$$\begin{aligned} \varphi_m &= \alpha * \varphi_{dom1} + (1-\alpha) * \varphi_{dom2} \\ \frac{\partial \varphi}{\partial n}_m &= \alpha * \frac{\partial \varphi}{\partial n}_{dom1} + (1-\alpha) * \frac{\partial \varphi}{\partial n}_{dom2} \end{aligned} \tag{3.10}$$

In Fig. 3.4 è riassunta la ricerca euristica del valore ottimale di θ. In questo caso è stato il valore $\theta = 0.6$ a fornire i risultati migliori.

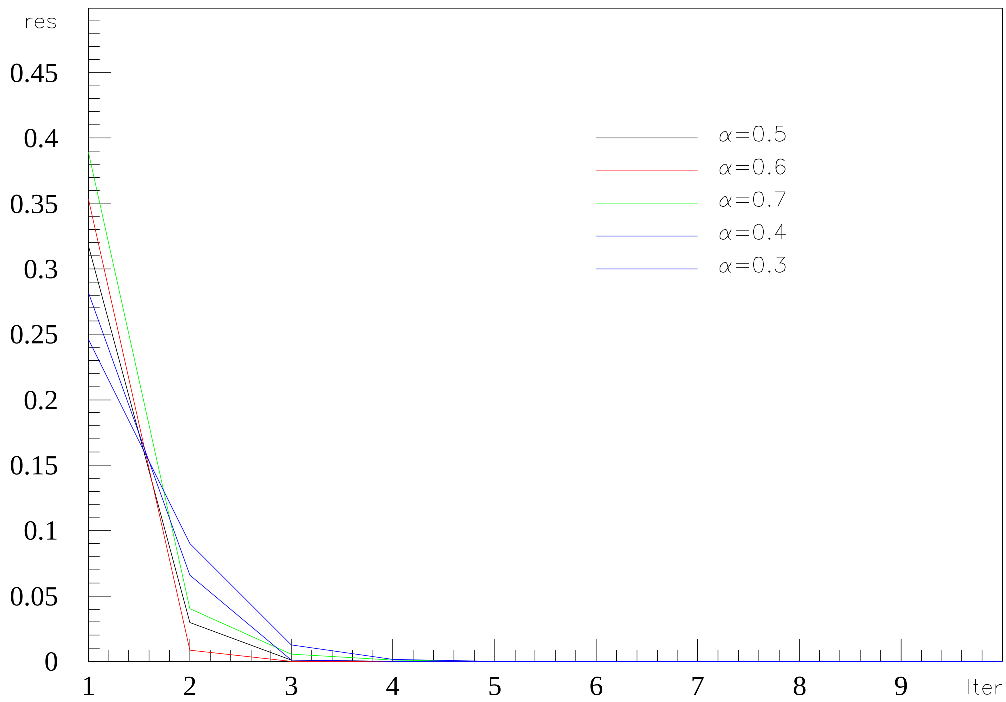

Figura 3.4: Ricerca euristica del valore ottimale di θ: metodo "misto".

Tale metodo, però, non presenta vantaggi rispetto a quello più semplice prima illustrato. Anche se apparentemente è necessario un minor numero di iterazioni per ridurre il residuo al di sotto di una soglia fissata, il tempo necessario è comunque pari a quello richiesto dal metodo precedente, considerato che per ogni iterazione, in questo caso, devono essere risolti due problemi attraverso due passi successivi. Pertanto nel seguito si è preferito fare riferimento al metodo Dirichlet-Neumann.

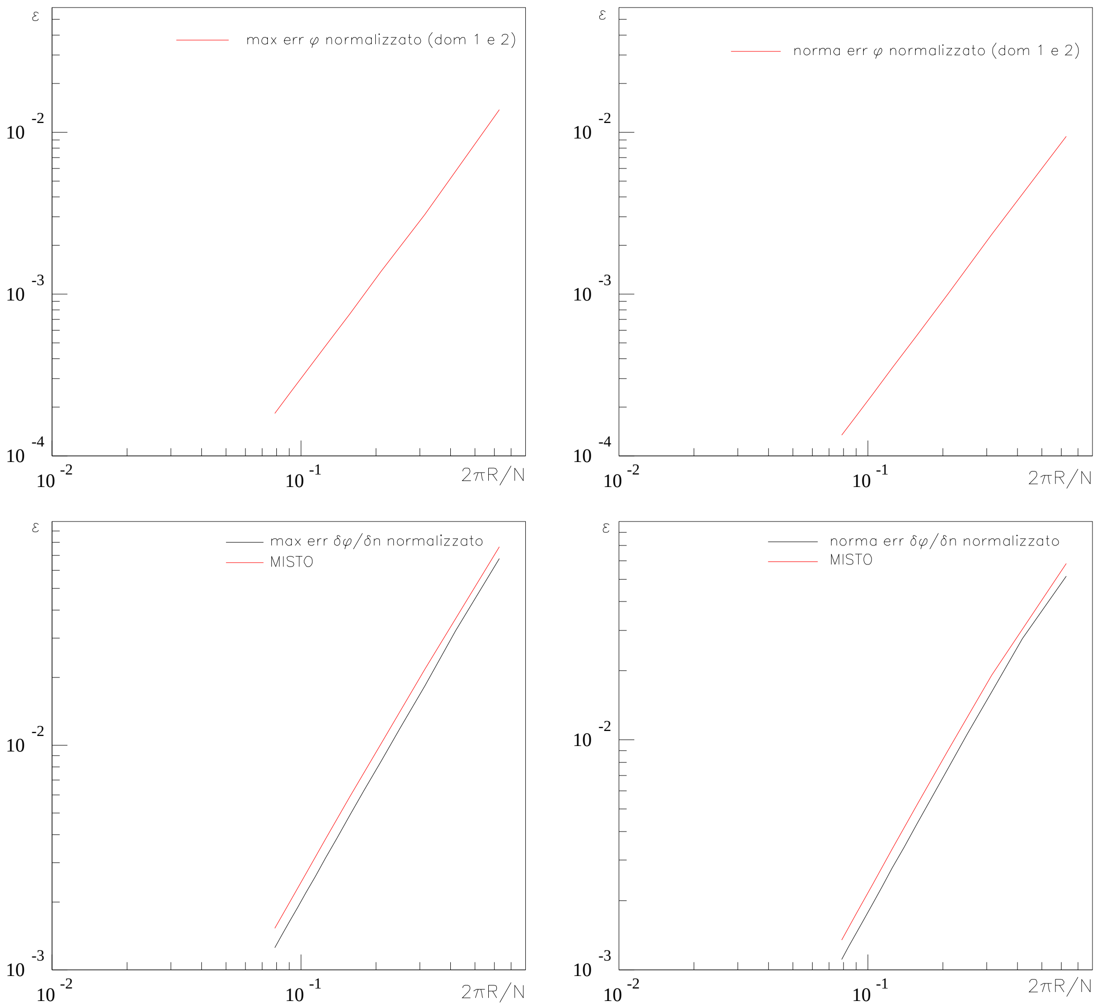

Figura 3.5: Errore al variare della dimensione dei pannelli utilizzati: griglia conforme, metodo misto.

Capitolo 4

Algoritmo FEM

Introduzione In questo capitolo, sono brevemente descritte le principali caratteristiche del codice FEM per la risoluzione del problema di Laplace in un dominio circolare. Per lo sviluppo di tale codice si è utilizzata la libreria **FEAT2D**-*Finite Element Analysis Tools*, sviluppata presso il Dipartimento di Matematica Applicata dell'Università di Heidelberg. Pertanto, si rimanda al relativo manuale per i dettagli relativi alle funzioni utilizzate ([1]).

Formulazione del problema Il problema ellittico descritto nei capitoli precedenti viene innanzitutto riformulato in forma debole, utilizzando la formulazione di *Galerkin*: sia dato il problema di Laplace

$$\begin{cases} \nabla^2\varphi & = & 0 & \text{in} & \mathring{\Omega} \\ \varphi & = & f & \text{su} & \mathcal{D} = \partial\Omega \end{cases} \tag{4.1}$$

Inizialmente si sono considerate condizioni al contorno di Dirichlet su tutta la frontiera. La formulazione debole si scrive:

$$\int_\Omega \nabla^2\varphi w = 0 \tag{4.2}$$

in cui w è una funzione test tale che

$$w \in V_0 = \left\{ u \in [H^1(\Omega)],\, u|_{\mathcal{D}} = 0 \right\} \tag{4.3}$$

Applicando il teorema di Green si ricava:

$$-\int_\Omega \nabla\varphi \cdot \nabla w + \int_{\partial\Omega} \frac{\partial\varphi}{\partial n} w = 0 \tag{4.4}$$

ovvero

$$\int_\Omega \nabla\varphi \cdot \nabla w = \int_{\mathcal{N}} \frac{\partial\varphi}{\partial n} w \tag{4.5}$$

dal momento che le funzioni test sono nulle sulla frontiera di Dirichlet. E poichè nel caso in esame su tutto il contorno vengono imposte condizioni di Dirichlet,

$$\int_{\Omega} \nabla \varphi \cdot \nabla w = 0 \tag{4.6}$$

Si introducono quindi le funzioni di forma $\mathcal{N}_k$, definite in modo tale che ciascuna di esse vale 1 nel nodo cui è associata e 0 in tutti gli altri nodi. In questo modo possiamo scrivere il potenziale φ come combinazione lineare delle funzioni di forma:

$$\varphi(P) = \sum_k \varphi_k \mathcal{N}_k(P) \tag{4.7}$$

Pertanto i valori ai nodi del potenziale φ_k diventano le nostre incognite. Analogamente il gradiente di φ sarà dato da

$$\nabla \varphi = \sum_k \varphi_k \nabla \mathcal{N}_k \tag{4.8}$$

E quindi si può scrivere:

$$\int_{\Omega} (\sum_k \varphi_k \nabla \mathcal{N}_k) \cdot \nabla w = 0 \tag{4.9}$$

La funzione test w viene scelta di volta in volta coincidente con una delle funzioni di forma, pertanto indicando con N il numero di incognite φ_k si avrà

$$\sum_k \varphi_k \int_{\Omega} \nabla \mathcal{N}_k \cdot \nabla \mathcal{N}_j = 0 \qquad \text{per} \quad j = 1, \ldots, N \tag{4.10}$$

in cui si è tenuto conto anche della linearità dell'operatore integrale. Di fatto l'integrale su Ω si traduce nella somme degli integrali sugli elementi e che costituiscono il supporto comune alle due funzioni di forma $\mathcal{N}_k$ e $\mathcal{N}_j$.

$$\sum_k \varphi_k \sum_e \int_{\Omega_e} \nabla \mathcal{N}_k \cdot \nabla \mathcal{N}_j = 0 \qquad \text{per} \quad j = 1, \ldots, N \tag{4.11}$$

In questo modo, si è giunti a riformulare il problema come un sistema lineare:

$$\mathbf{A} \star \varphi = 0 \tag{4.12}$$

in cui φ è il vettore delle incognite, mentre $\mathbf{A}$ è detta *stiffness matrix*.

Codice FEM: struttura Il codice implementato si articola nei seguenti passi:

1. Generazione della griglia di calcolo

2. Assemblaggio della stiffness matrix
3. Assemblaggio del vettore termine noto (in questo primo caso i termini noti sono nulli, ma è comunque previsto che possano essere diversi da zero.)
4. Implementazione delle condizioni al contorno
5. Soluzione del sistema lineare con il metodo del Gradiente Coniugato
6. Analisi degli errori

Al termine di questo processo, si ha una fase di *post-processing* necessaria per il calcolo delle derivate del potenziale e per la scrittura di file di output per pacchetti grafici (Tecplot).

Generazione della griglia di calcolo Il dominio viene descritto dall'utente attraverso una griglia grossolana. Tale suddivisione viene poi infittita automaticamente tramite funzioni implementate in **FEAT**, in base all'indicazione del numero di raffinamenti che devono essere eseguiti. Nel caso in esame, il dominio circolare è stato suddiviso uniformemente in elementi triangolari.

Assemblaggio della stiffness matrix Sfruttando la struttura della stiffness matrix prima descritta, tale matrice viene costruita nel modo seguente. Innanzitutto, nel caso di elementi triangolari risulta conveniente utilizzare "coordinate areali" (vedi [6]): per un generico punto P, le coordinate saranno

$$L_1 = \frac{A_1}{A}, \qquad L_2 = \frac{A_2}{A}, \qquad L_3 = \frac{A_3}{A} \tag{4.13}$$

dove A è l'area del triangolo e A_1, A_2, A_3 le aree in cui il triangolo resta suddiviso dal punto P, opposte al nodo omonimo. Chiaramente, le coordinate L_i sono legate a quelle cartesiane:

$$L_i = \frac{a_i + b_i x + c_i y}{2A} \tag{4.14}$$

dove, ad esempio per $i = 1$

$$\begin{aligned} a_1 &= x_2 y_3 - x_3 y_2 \\ b_1 &= y_2 - y_3 \qquad (4.15) \\ c_1 &= x_3 - x_2 \qquad (4.16) \end{aligned}$$

mentre gli altri coefficienti si ottengono permutando opportunamente gli indici. Corrispondentemente, le derivate valgono

$$\frac{\partial L_i}{\partial x} = \frac{b_i}{2A} \tag{4.17}$$

$$\frac{\partial L_i}{\partial y} = \frac{c_i}{2A} \tag{4.18}$$

Le L_i possono essere impiegate come funzioni di forma e pertanto il generico elemento della stiffness matrix è dato da

$$\begin{aligned} k_{ij} &= \int_A \frac{\partial L_i}{\partial x}\frac{\partial L_j}{\partial x} + \frac{\partial L_i}{\partial y}\frac{\partial L_j}{\partial y} \\ &= \int_A \frac{b_i b_j}{4A^2} + \frac{c_i c_j}{4A^2} \end{aligned} \tag{4.19}$$

Di fatto, per ogni elemento viene costruita una stiffness matrix locale (considerando solo i nodi appartenenti all'elemento in esame) e poi i coefficienti vengono assemblati nella stiffness matrix globale. Sia le matrici elementari che quella globale sono simmetriche.

Assemblaggio del vettore termine noto Nel caso in esame tale vettore è nullo. Pertanto per i dettagli relativi all'introduzione di un termine noto diverso da zero si veda il manuale [1]. Successivamente verrà affrontato il problema delle forme lineari quando di dovrà tener conto anche di condizioni al contorno di Neumann.

Implementazione delle condizioni al contorno Nella fase di assemblaggio della stiffness matrix, quest'ultima matrice viene costruita senza tener conto delle condizioni al contorno. Pertanto come incognite compaiono i valori del potenziale in tutti i nodi della discretizzazione e la matrice A è una matrice $N \times N$. In questa fase, si deve pertanto tener conto delle condizioni al contorno di Dirichlet sulla frontiera. Ciò si ottiene tramite una funzione definita dall'utente, che consiste in alcuni passi successivi: per ogni punto del contorno

- Il corrispondente elemento diagonale nella stiffness matrix viene posto uguale a 1;
- Tutti gli altri elementi della stessa riga vengono posti uguali a zero;
- Il termine noto ed l'incognita corrispondente vengono posti direttamente uguali al valore noto del potenziale.

Soluzione ed analisi degli errori Una volta costruito il sistema lineare, questo viene risolto con un metodo iterativo. Tra quelli resi disponibili nella libreria di **FEAT**, si è scelto il Gradiente Coniugato precondizionato (Precondizionamento SSOR con parametro di rilassamento $\omega = 1$). Successivamente, si è effettuata un'analisi degli errori attraverso il calcolo della norma-∞ e della norma-2. Tale analisi ha dimostrato come il calcolo del potenziale sia piuttosto accurato e coma l'accuratezza cresca abbastanza rapidamente all'infittirsi della griglia (Figg. 4.1, 4.2). Si può anche notare come non sia conveniente aumentare inizialmente il numero di punti per la descrizione del contorno, anche perchè l'infittimento comporta anche l'aumenta del numero di nodi sul contorno.
Si noti anche che l'analisi degli errori, così effettuata, è responsabile dei lunghi tempi di calcolo richiesti dal codice. Una volta eliminato il calcolo degli errori, il codice è risultato piuttosto efficiente.

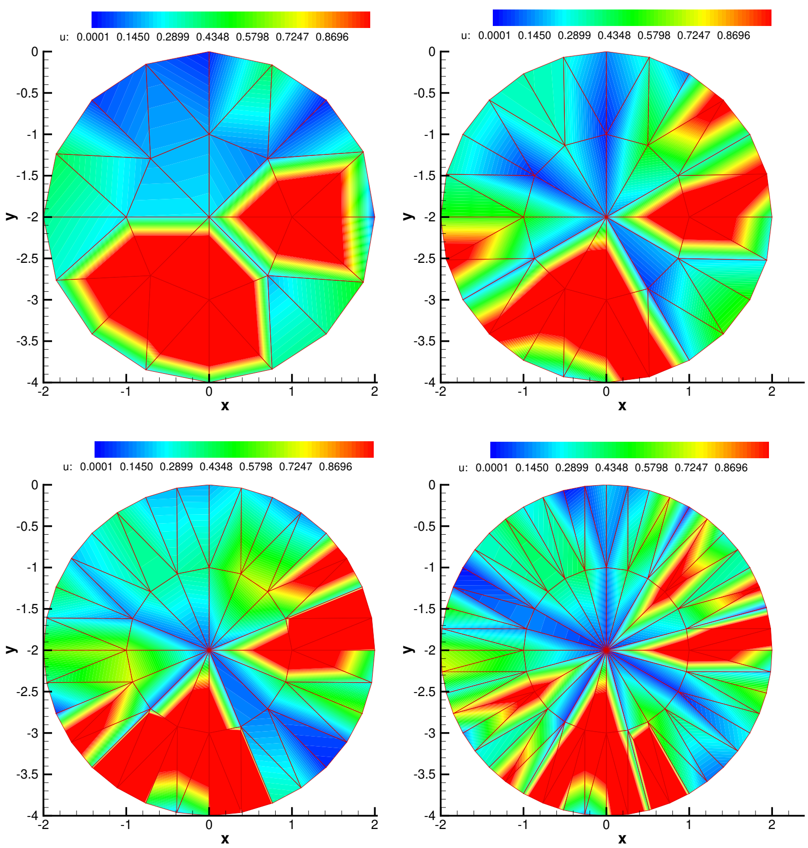

Figura 4.1: Errore relativo nella valutazione del potenziale per il caso test considerato (dominio circolare di raggio 2, $\varphi = \exp ky \sin kx$ con $k = 1$. L'errore varia da 0 (blu) a 1 o valori maggiori[1] di 1 (rosso). Variazione dell'errore all'aumentare del numero di punti utilizzati per la descrizione del contorno.

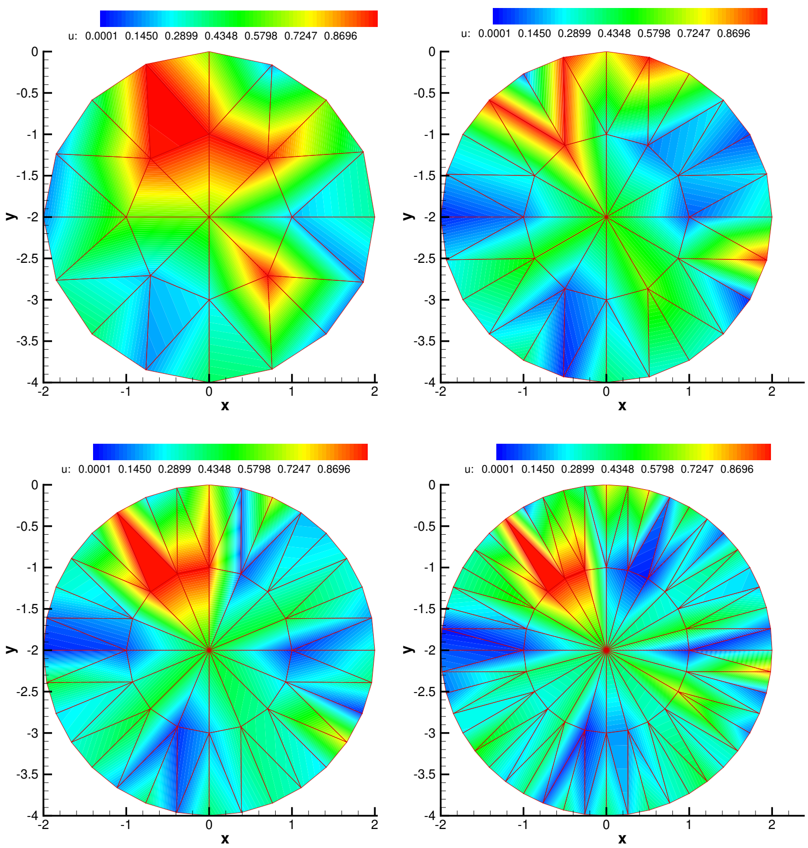

Figura 4.2: Errore relativo nella valutazione del potenziale per il caso test considerato (dominio circolare di raggio 2, $\varphi = \exp ky \sin kx$ con $k = 1$. L'errore varia da 0 (blu) a 1 o valori maggiori di 1 (rosso). Variazione dell'errore all'aumentare del numero di raffinamenti.

Capitolo 5

Algoritmo FEM: calcolo delle derivate al contorno

Introduzione Il problema della determinazione di quantità derivate dalle variabili primarie ricavate con un metodo ad elementi finiti è un problema di non facile soluzione. In generale la semplice derivazione delle funzioni di forma produce risultati poco accurati. Pertanto sono stati proposti in letteratura diversi metodi per il calcolo delle grandezze derivate in corrispondenza dei nodi: molti di questi si basano su una media opportunamente pesata dei valori delle variabili primarie. Il metodo ottimale dipende comunque dal tipo di grandezza che si deve ricavare: una rassegna esauriente di tali metodi si trova in [7].

Nel nostro caso, si trattava di calcolare le derivate normali del potenziale sul contorno del dominio. La soluzione di tale problema era essenziale dal momento che, nel caso del flusso irrotazionale di un liquido incomprimibile, il gradiente del potenziale rappresenta la velocità e, dunque, è necessario calcolarlo se si vuole determinare, ad esempio, l'evoluzione della superficie libera nel tempo. Inoltre, in vista dell'accoppiamento con l'algoritmo BEM e scelto il metodo Dirichlet-Neumann come metodo di accoppiamento, è necessario saper calcolare con adeguata precisione le derivate del potenziale sul contorno del dominio (ovvero sulla frontiera di trasmissione).
Nel presente capitolo sono riportati i metodi impiegati per il calcolo delle derivate ed i risultati ottenuti. Come si vedrà, i risultati migliori sono quelli relativi al metodo detto "Superconvergent Patch Recovery", dovuto a Zienkiewicz e Zhu. Tale metodo è stato dunque impiegato, nel programma di accoppiamento FEM-BEM, per la determinazione delle derivate del potenziale ricavato con il codice FEM.

Calcolo delle derivate: derivate delle funzioni di forma In un primo tempo, le derivate del potenziale sono state calcolate tramite le derivate delle funzioni di forma utilizzate per il potenziale stesso. La conoscenza delle derivate delle funzioni di forma rappresenta, comunque, un primo passo anche per l'applicazione di metodi più sofisticati. Come la letteratura faceva presagire (vedi [7]), i risultati non sono particolarmente soddisfacenti: effettivamente, la soluzione converge alla soluzione analitica al diminuire della dimensione caratteristica della griglia, ma tale convergenza è piuttosto lenta. Per ottenere una valutazione delle derivate sufficientemente precisa è necessario utilizzare griglie molto fitte con tempi di calcolo conseguentemente troppo alti in relazione al nostro obiettivo di ottenere l'evoluzione temporale del dominio fluido. È opportuno precisare, però, che si è successivamente verificato che gli elevatissimi tempi di calcolo erano dovuti essenzialmente al calcolo dell'errore: infatti le subroutine corrispondenti sono piuttosto dispendiose ed eliminandole si ottiene un enorme risparmio in termini di tempo di calcolo.

Il limite maggiore di questo procedimento, nel caso specifico, era rappresentato dal fatto che gli elementi utilizzati sono lineari e pertanto le derivate delle funzioni di forma risultano costanti su ciascun elemento. Si presentava dunque il problema di capire quale valore attribuire alla derivata sul contorno. Per ridurre l'imprecisione relativa alla scelta della normale, il valore ottenuto per gli elementi di bordo è stato attribuito al punto medio dello spigolo giacente sul contorno e, come normale, si è considerata la normale a tale spigolo.

Tale procedimento è stato quindi applicato al problema prototipo descritto nel capitolo 2.

La derivata normale veniva calcolata nel modo seguente:

$$\frac{\partial \phi}{\partial n} = \frac{\partial \phi}{\partial x} * n_x + \frac{\partial \phi}{\partial y} * n_y \tag{5.1}$$

Come già preannunciato, i valori ottenuti in questo modo rappresentavano una pessima approssimazione delle derivate analitiche. Infatti così come gli errori sul potenziale (Fig. 4.1 e 4.2 nel cap. 4), anche gli errori sulle derivate parziali del potenziale si riducono all'infittirsi della griglia ma in modo estremamente lento. Inoltre, come si vede in Fig. 5.1, gli errori restano elevati in alcune zone del dominio anche per griglie molto fitte. Ciò si ripercuote negativamente sul valore della derivata normale del potenziale.

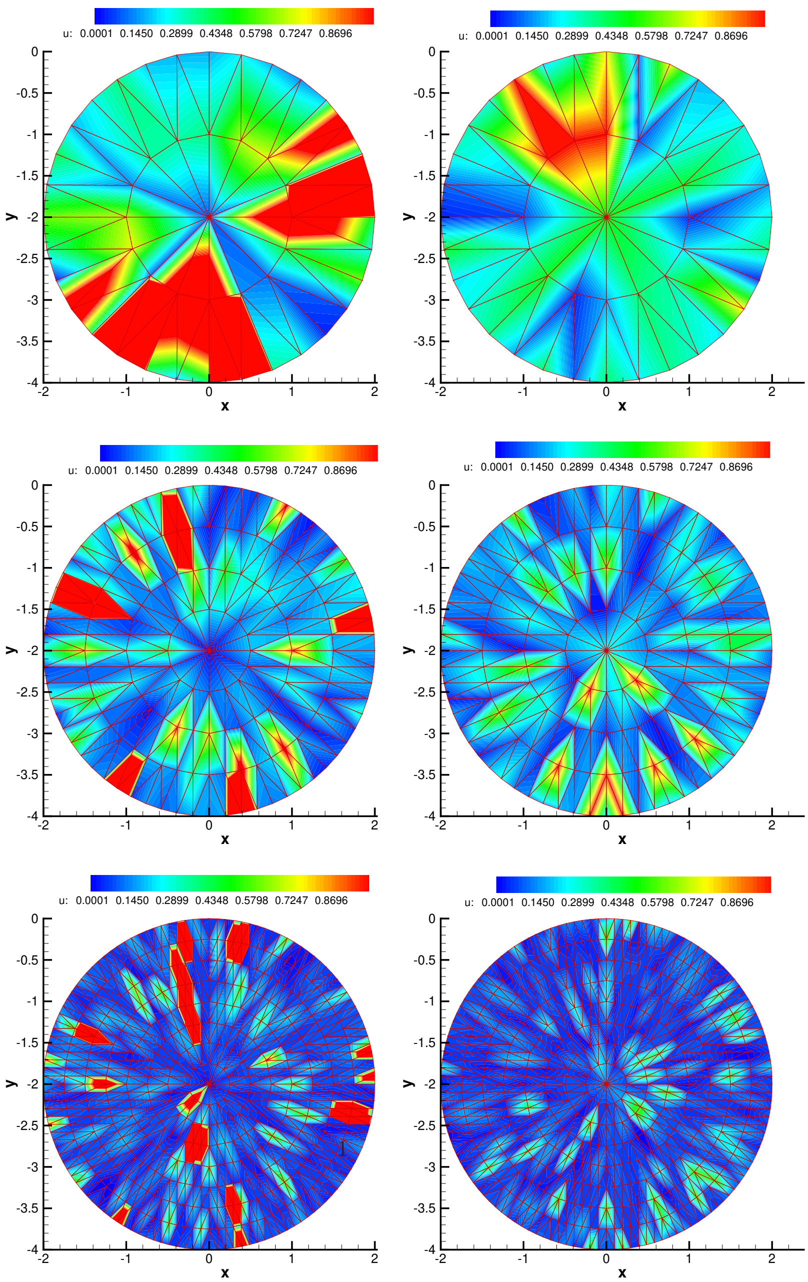

Figura 5.1: Errore relativo: a sinistra $\frac{\partial \varphi}{\partial x}$, a destra $\frac{\partial \varphi}{\partial y}$. Caso test: dominio circolare di raggio 2, $\varphi = \exp ky \sin kx$ con $k = 1$. L'errore varia da 0 (blu) a 1 o valori maggiori di 1 (rosso).

I risultati ottenuti con la griglia più fitta (le cui dimensioni caratteristiche sono tuttavia compatibili con quelle riportate in vari articoli, ad es. [16]) sarebbero piuttosto soddisfacenti se non comportassero tempi di calcolo eccessivamente elevati. Il problema dei tempi di calcolo è acuito dal fatto che nel codice finale tale calcolo dovrà essere eseguito per ogni intervallo di tempo e pertanto il tempo di calcolo per ciascun time step deve essere ridotto il più possibile.

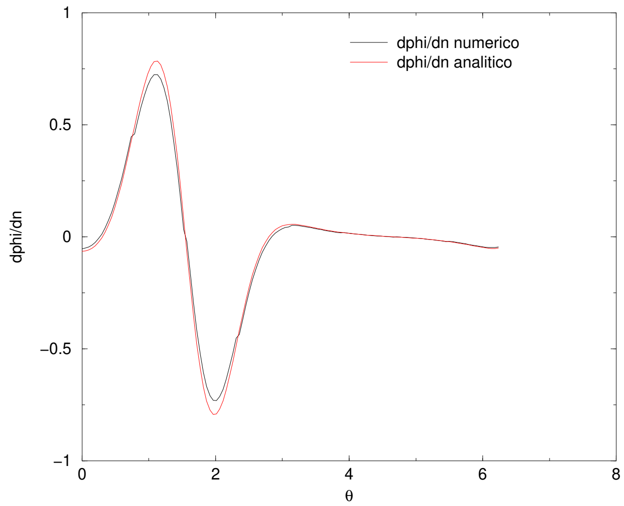

Figura 5.2: Andamento della $\frac{\partial\phi}{\partial n}$ sul contorno: dimensione caratteristica degli elementi $h = 0.125$. Confronto tra il valore analitico ed il risultato numerico.

Si poneva, pertanto, il problema di trovare un algoritmo efficace per il calcolo delle derivate normali al contorno. Gli autori non sono giunti ad argomentazioni davvero conclusive in proposito. In [16], sono mostrati i risultati di simulazioni di sloshing con metodi ad elementi finiti , ma non è precisato il modo in cui sono ottenute le derivate al contorno. In [3], si utilizza un trasformazione del dominio fisico (*σ-transformation*) che evita la rigrigliatura ad ogni time step; la griglia nel dominio trasformato è regolare e ciò semplifica il calcolo delle derivate. Tale soluzione è stata però scartata in quanto, a nostro parere, impedirebbe di seguire l'eventuale frangimento delle onde. Anche [4] rimanda alla σ-transformation, mentre in [2] si parla di non meglio precisati "accurati schemi di interpolazione di ordine elevato". In un primo tempo, si è pensato di implementare uno schema di interpolazione che desse risultati più precisi rispetto alla semplice derivazione delle funzioni di forma.

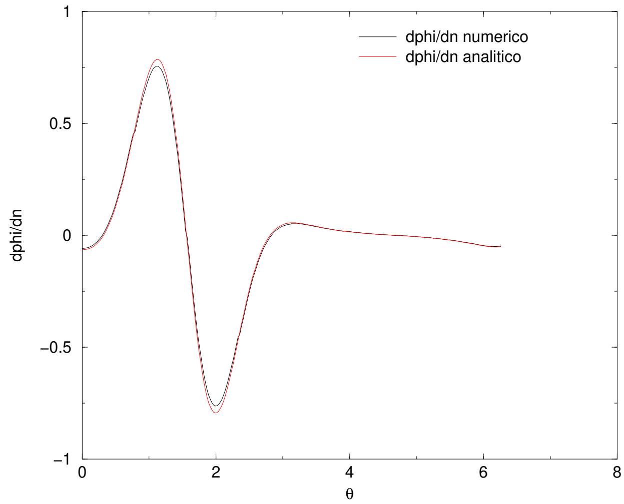

Figura 5.3: Andamento della $\frac{\partial \phi}{\partial n}$ sul contorno: dimensione caratteristica degli elementi $h = 0.0625$. Confronto tra il valore analitico ed il risultato numerico.

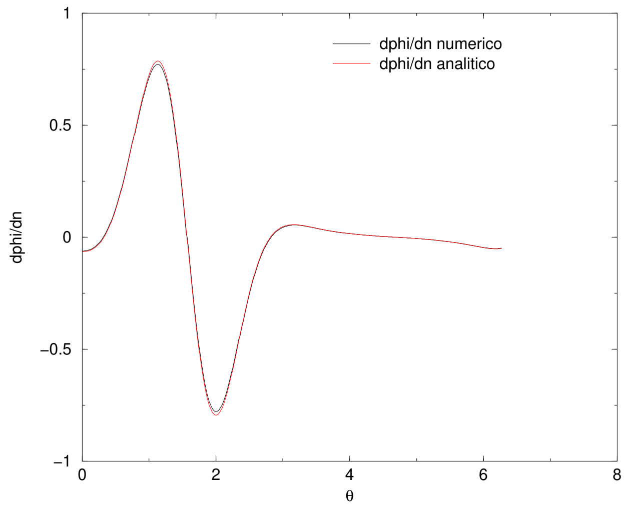

Figura 5.4: Andamento della $\frac{\partial \phi}{\partial n}$ sul contorno: dimensione caratteristica degli elementi $h = 0.03125$. Confronto tra il valore analitico ed il risultato numerico.

Calcolo delle derivate con una formula down-wind del primo ordine
L'idea di fondo era quella di riuscire a calcolare le derivate sfruttando anche l'informazione relativa agli elementi più interni rispetto alla frontiera. A questo scopo, si è pensato di impiegare un formula *down-wind* del primo ordine per il calcolo delle derivate normali. Nel caso del cerchio, la direzione normale coincide con quella radiale e pertanto, dati tre punti allineati lungo un raggio (di cui il punto 1 sulla frontiera, e gli altri due equispaziati verso il centro), la derivata normale è stata calcolata come:

$$f_1' = \frac{\frac{1}{\Delta x_2{}^2}\,(f_2 - f_1) - \frac{1}{\Delta x_3{}^2}\,(f_3 - f_1)}{\left(\frac{1}{\Delta x_3} - \frac{1}{\Delta x_2}\right)} \tag{5.2}$$

dove Δx_2 rappresenta la distanza tra il punto 1 e il punto 2, mentre Δx_3 rappresenta la distanza tra il punto 1 e il punto 3. Con questa formula, è possibile calcolare la derivata direttamente dai valori del potenziale ai nodi senza ricorre alle derivate delle funzioni di forma.

Innanzitutto sono stati costruiti tre vettori contenenti, rispettivamente, i nodi sulla frontiera e quelli appartenenti alle due circonferenze immediatamente più interne. I nodi dei triangoli ricadono, infatti, su circonferenze concentriche (vedi Fig. 5.1, per es.). Quindi, per ogni punto sulla frontiera, si devono individuare i due punti interni corrispondenti, allineati sullo stesso raggio; quando non esisteva un nodo allineato sul raggio, si consideravano i due nodi immediatamente precedente e seguente. In quest'ultimo caso il valore del potenziale da introdurre nella formula 5.2 veniva ricavato come media tra i valori nei due nodi selezionati. Chiaramente questo introduce un ulteriore elemento di approssimazione, ma tale errore dovrebbe comunque ridursi all'infittirsi della griglia utilizzata.

Come si vede in Fig.5.5, i risultati ottenuti, tuttavia, sono stati tutt'altro che incoraggianti e non migliorano infittendo ulteriormente la griglia.

È stata fatta anche un'ulteriore verifica con un dominio rettangolare (Larghezza=2, Altezza=1). Sebbene in questo caso i risultati siano più soddisfacenti (Fig. 5.6) rispetto al caso del dominio circolare, l'algoritmo non risulta affatto efficiente. Inoltre la precisione non è superiore a quella ottenuta derivando le funzioni di forma.

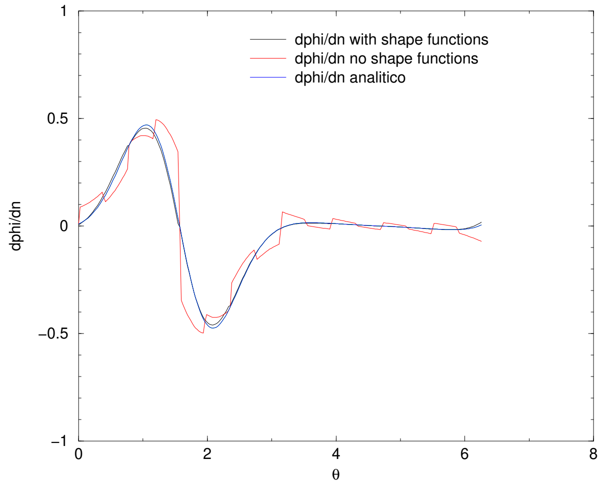

Figura 5.5: Derivata normale sul contorno: caso del dominio circolare, $\varphi = \exp ky \sin kx$. Dimensione caratteristica della griglia 0.0625.

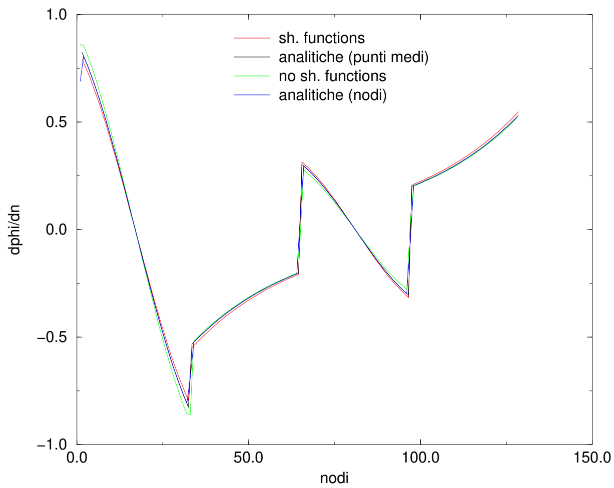

Figura 5.6: Derivata normale sul contorno: caso del dominio rettangolare, $\varphi = \exp ky \sin kx$. Dimensione caratteristica della griglia 0.0625.

"Patch Recovery Method" Questo metodo, definito in [7] come il metodo probabilmente migliore per calcolare le derivate della funzione incognita ai nodi, si basa sull'idea di sfruttare la superiore accuratezza o "superconvergenza" caratteristica di alcuni punti degli elementi della discretizzazione. Tale metodo è illustrato e verificato in una serie di articoli di Zienkiewicz e Zhu ([18], [19], [17]), ed è comunemente noto con il nome di "metodo Z^2". La procedura si basa su un'espansione polinomiale continua della funzione che descrive le derivate: tale polinomio è ricavato per un insieme di elementi (detto *patch*) che circonda il nodo in corrispondenza del quale si desidera calcolare la derivata. Tale polinomio viene ricavato sfruttando i valori delle derivate in corrspondenza dei punti di super-convergenza ed interpolandoli con un metodo ai minimi quadrati o con una proiezione L^2. Secondo Zienkiewicz e Zhu, il metodo basato sui minimi quadrati conduce sempre al recupero super-convergente delle derivate ai nodi e pertanto è stato utilizzato nel codice. In [18] è anche detto che tale metodo garantisce un'accuratezza di un ordine superiore nel caso delle derivate di elementi lineari e cubici e di due ordini di superiore nel caso di elementi quadratici.
La procedura può essere così schematizzata ([7]):

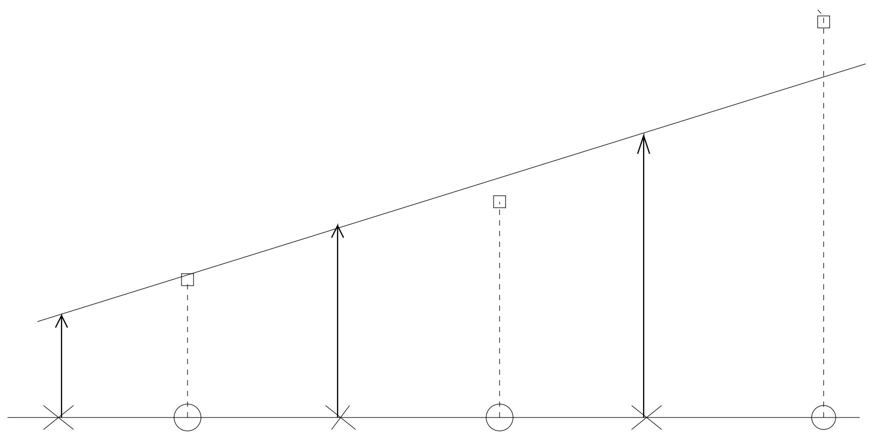

Figura 5.7: Schema della procedura di Patch Recovery: caso 1D, elementi lineari.

1. Si calcolano le derivate della soluzione ad elementi finiti nei punti di superconvergenza; queste derivate sono calcolate utilizzando le derivate delle funzioni di forma. In Fig. 5.7, è riportato un esempio 1D con elementi lineari: vengono valutate le derivate come derivate delle funzioni di forma (quadrati) in corrispondenza dei punti medi degli elementi

(cerchi). Nel caso degli elementi triangolari lineari, l'esistenza e la locazione dei punti di superconvergenza sono ancora un problema che non sembra essere stato pienamente risolto matematicamente; sembra, comunque, che nel caso di elementi lineari si possa fare riferimento ai baricentri. È stato comunque verificato che l'accuratezza in corrispondenza dei baricentri è superiore a quella che si ha in corrispondenza dei nodi (Fig. 5.8).

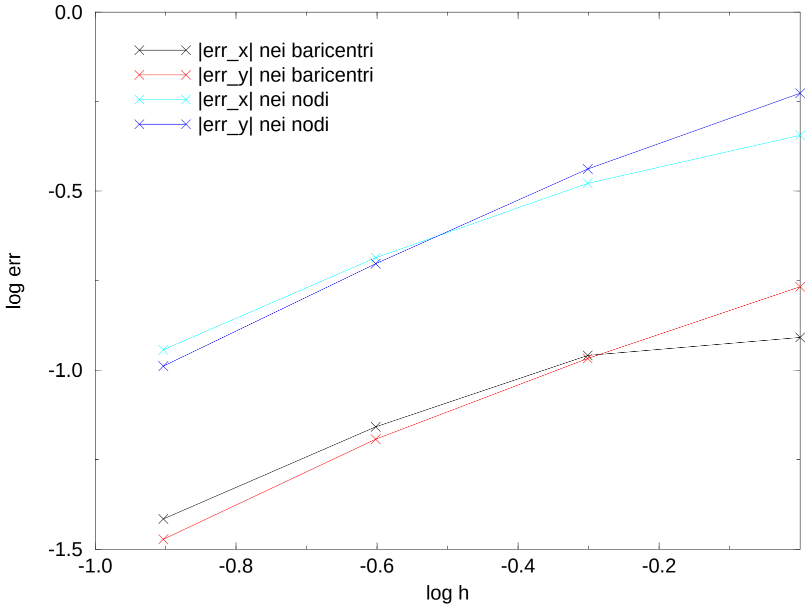

Figura 5.8: Velocità di convergenza per elementi linerari: confronto tra i nodi ed i baricentri degli elementi.

2. Si calcola il polinomio interpolante con un metodo di adattamento ai minimi quadrati. Detto σ la derivata della funzione e σ_h la derivata valutata tramite le derivate delle funzioni di forma, l'obiettivo è trovare dei valori nodali $\overline{\sigma}^*$, tali che le derivate

$$\sigma^* = \mathbf{N}\overline{\sigma}^* \tag{5.3}$$

siano più accurate delle σ_h. $\mathbf{N}$ sono le funzioni di forma. I valori nodali $\overline{\sigma}^*$ si ricavano da un'espansione polinomiale σ_p^* dello stesso ordine p delle funzioni di base $\mathbf{N}$ valido su una patch che circonda il nodo considerato.

$$\sigma_p^* = \mathbf{Pa} \tag{5.4}$$

dove **P** contiene i termini polinomiali appropriati e **a** è un set di parametri incogniti. Nel nostro caso:

$$\mathbf{P} = [1 \qquad x \qquad y] \tag{5.5}$$

$$\mathbf{a} = [a_0 \qquad a_1 \qquad a_2]^T \tag{5.6}$$

La determinazione dei parametri incogniti **a** si ottiene mediante un adattamento ai minimi quadrati utilizzando i valori "super-convergenti": si tratta cioè di minimizzare la funzione

$$\begin{aligned}\mathbf{F}(\mathbf{a}) &= \sum_{i=1}^{n}(\sigma_h(x_i,y_i) - \sigma_p^*(x_i,y_i))^2 \\ &= \sum_{i=1}^{n}(\sigma_h(x_i,y_i) - \mathbf{P}(x_i,y_i)\mathbf{a})^2\end{aligned} \tag{5.7}$$

dove (x_i, y_i) sono le coordinate dei punti di superconvergenza (nel nostro caso i baricentri dei triangoli). La condizione di minimizzazione implica che **a** soddisfi il sistema

$$\sum_{i=1}^{n}\mathbf{P}^T(x_i,y_i)\mathbf{P}(x_i,y_i)\mathbf{a} = \sum_{i=1}^{n}\mathbf{P}^T(x_i,y_i)\sigma_h(x_i,y_i) \tag{5.8}$$

3. Si calcolano i nuovi valori nodali valutando il polinomio risultante in corrispondenza dei nodi (X).

$$\overline{\sigma}^*(x_n, y_n) = a_0 + a_1 * x_n + a_2 * y_n \tag{5.9}$$

Un'attenzione particolare merita la valutazione delle derivate in corrispondenza dei nodi di bordo (come nel nostro caso), in quanto ciascuna patch può essere costituita soltanto da uno o due elementi. Nel caso che la patch sia costituita da un solo elemento, chiaramente la dimensione della patch è insufficiente per la determinazione di **a** ed è quindi necessario ricorre ad una patch più interna che contenga il nodo in esame. Per patch costituite da due elementi, si può utilizzare il metodo standard. In generale, risulta però più conveniente utilizzare patch centrate sui nodi più interni. Questa procedura è stata seguita nel codice sviluppato.

Con questo procedimento, i risultati ottenuti sono risultati estremamente soddisfacenti. In Fig. 5.9, si può notare come tale metodo consenta un'ottima approssimazione delle derivate parziali nel centro delle patch anche con un numero di raffinamenti, pari a 4, relativamente basso. Con lo stesso numero di raffinamenti, si ottiene (molto rapidamente) anche un'approssimazione accettabile della derivata normale in corrispondenza del contorno (Fig.5.10). Analogamente, si sono ottenuti risultati soddisfacenti considerando la funzione $\varphi = \exp 3y \sin 3x$ (Fig. 5.11).

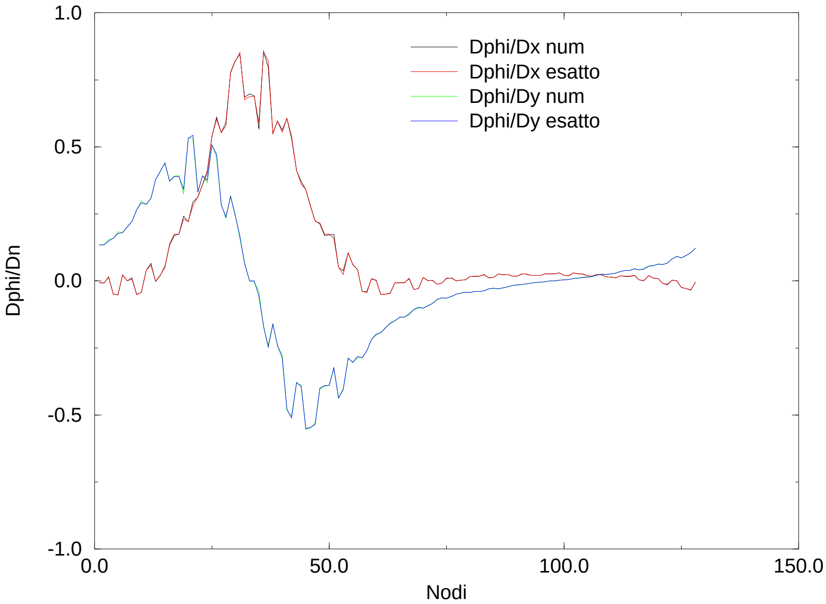

Figura 5.9: Approssimazione delle derivate parziali in corrispondenza dei centri delle patch con il metodo Patch Recovery: caso del dominio circolare, $\varphi = \exp y \sin x$, numero di raffinamenti=4.

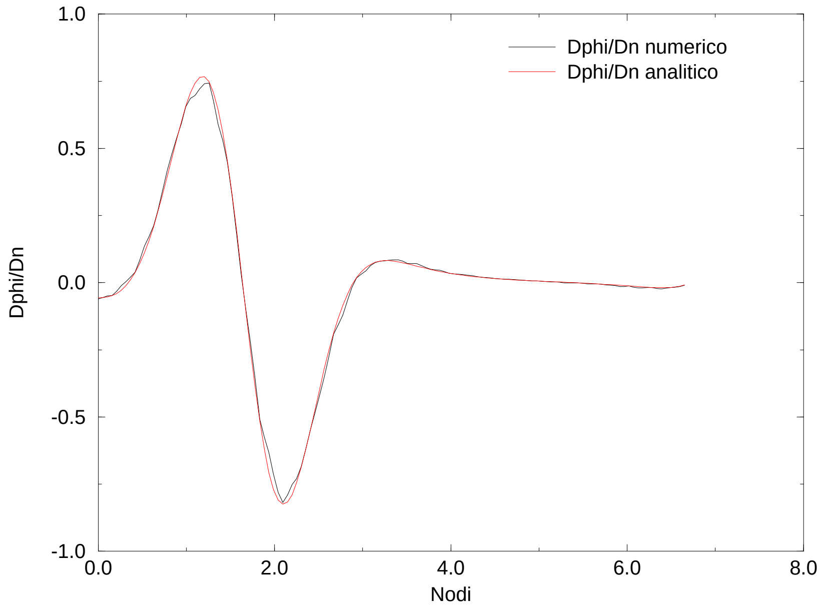

Figura 5.10: Derivate normali in corrispondenza del contorno con il metodo Patch Recovery: caso del dominio circolare, $\varphi = \exp y \sin x$, numero di raffinamenti=4.

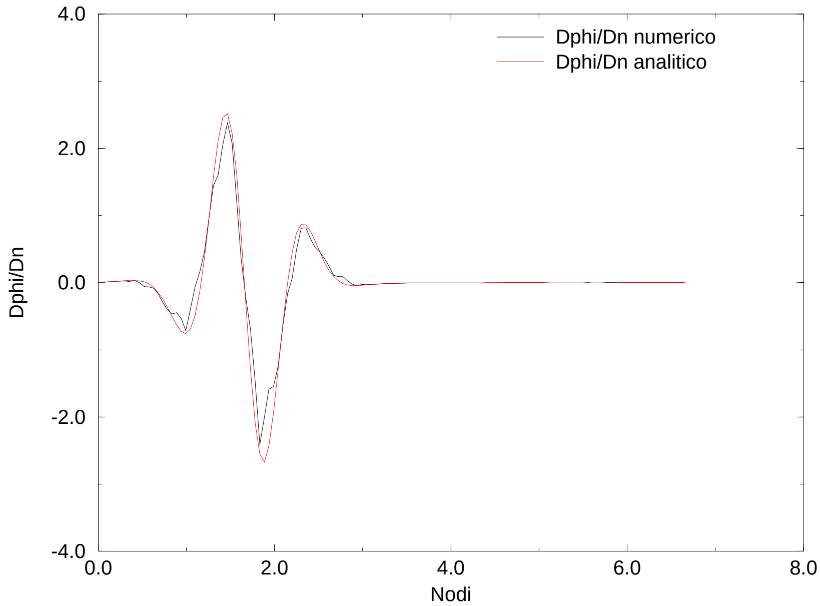

Figura 5.11: Derivate normali in corrispondenza del contorno con il metodo Patch Recovery: caso del dominio circolare, $\varphi = \exp 3y \sin 3x$, numero di raffinamenti=4.

Dominio rettangolare Poichè i risultati relativi al dominio circolare sono risultati molto soddisfacenti, si è passati ad un dominio rettangolare di lunghezza pari a 2 e altezza pari ad 1. Questo caso è più vicino a quello che si dovrà considerare nel codice finale, dal momento che il dominio fluido (a parte la presenza del corpo galleggiante) verrà suddiviso in un dominio rettangolare interno ed uno più esterno. In questo caso, i risultati sono stati ancora migliori di quelli ottenuti nel caso del dominio circolare. Nelle Figg. 5.12, 5.13, 5.14 è evidente come, con un numero di raffinamenti estremamente basso (3), si possa ottenere un'ottima approssimazione delle derivate normali al contorno anche con funzioni via via più "accidentate".

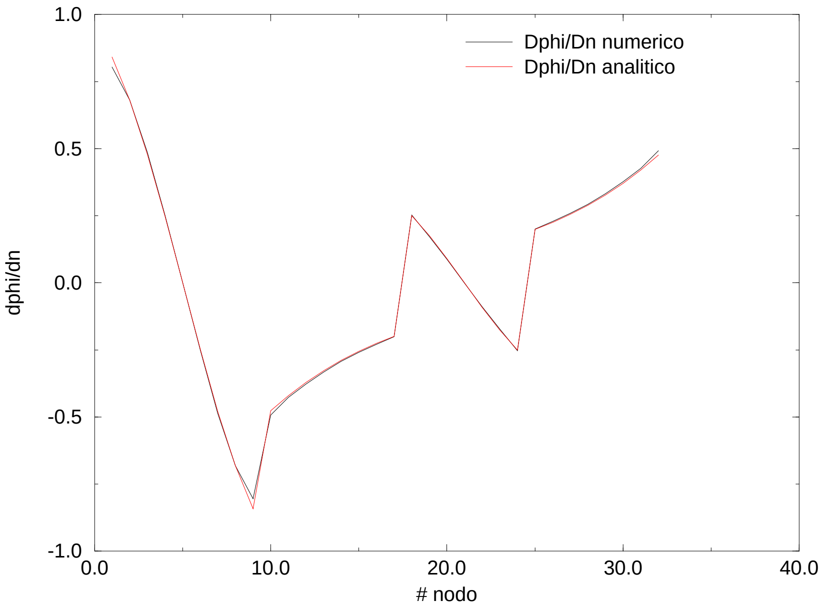

Figura 5.12: Derivate normali al contorno con il metodo Patch Recovery: caso del dominio rettangolare, $\varphi = \exp y \sin x$, numero di raffinamenti=3.

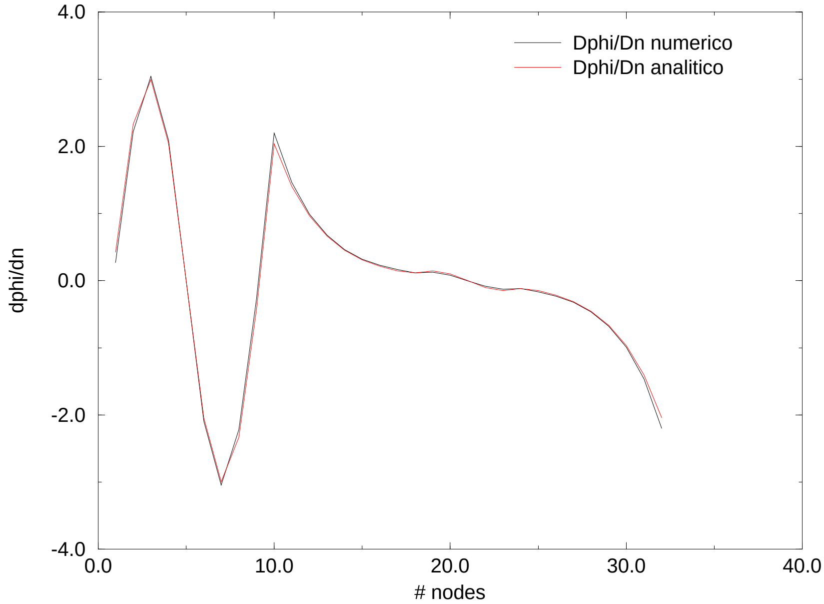

Figura 5.13: Derivate normali al contorno con il metodo Patch Recovery: caso del dominio rettangolare, $\varphi = \exp 3y \sin 3x$, numero di raffinamenti=3.

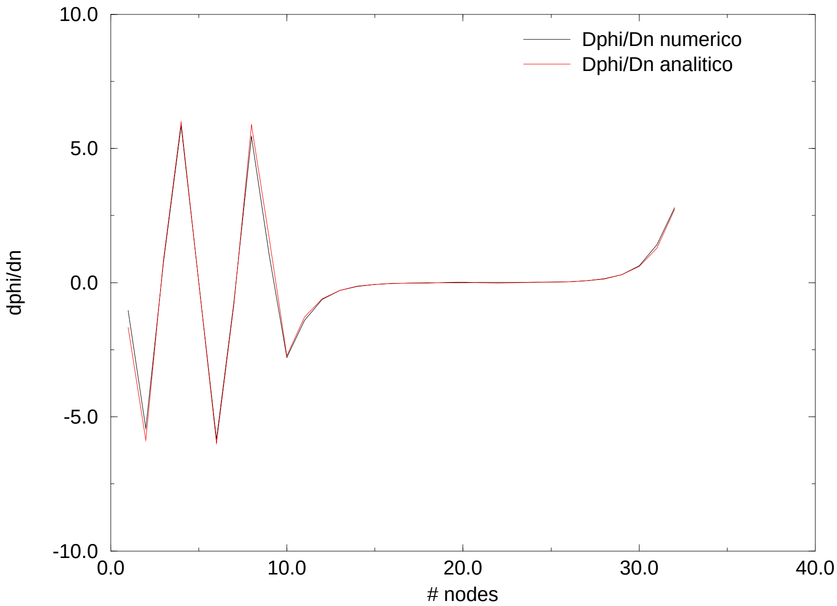

Figura 5.14: Derivate normali al contorno con il metodo Patch Recovery: caso del dominio rettangolare, $\varphi = \exp 6y \sin 6x$, numero di raffinamenti=3.

Nuove condizioni al contorno Il passo successivo da compiere era quello di implemetare delle condizioni al contorno miste: **Dirichlet su una parte del contorno e Neumann sulla parte restante**. Infatti, nel problema reale, mentre sulla superficie libera del fluido si imporranno sempre delle condizioni tipo Dirichlet, sugli altri lati del dominio si possono avere condizioni tipo Neumann, sia per il tipo di algoritmo di accoppiamento scelto per la domain decomposition sia perchè i lati possono coincidere con delle pareti rigide. Chiaramente, l'introduzione di condizioni al contorno tipo Neumann modifica la formulazione del problema: si riprende la formula 4.5

$$\int_{\Omega} \nabla\varphi \cdot \nabla w = \int_{\mathcal{N}} \frac{\partial\varphi}{\partial n} w \tag{5.10}$$

In questo caso, la forma lineare a secondo membro è in generale diversa da zero. Pertanto la 4.11 diventa:

$$\sum_k \varphi_k \sum_e \int_{\Omega_e} \nabla\mathcal{N}_k \cdot \nabla\mathcal{N}_j = \sum_{N_e} \int_{N_e} \frac{\hat{\partial\varphi}}{\partial n} \cdot \mathbf{n}_e \mathcal{N}_j \qquad \text{per} \quad j = 1, \ldots, N \tag{5.11}$$

Per il calcolo della forma lineare, **FEAT** fornisce una funzione che effettua la cubatura analoga a quella utilizzata per il calcolo delle forme bilineari. Le normali e i coefficienti $\frac{\hat{\partial\varphi}}{\partial n}$, però, devono essere precedentemente calcolati in base alla geometria del sistema e alla funzione imposta (nel caso test sempre $\varphi = \exp y \sin x$). Anche in questo caso, si sono ottenuti risultati molto soddisfacenti anche con numeri di raffinamento bassi. Le Fig. 5.15 e 5.16 si riferiscono al caso con numero di raffinamento 5: in questo caso, l'approssimazione ottenuta è ottima e i tempi di calcolo sufficientemente bassi.

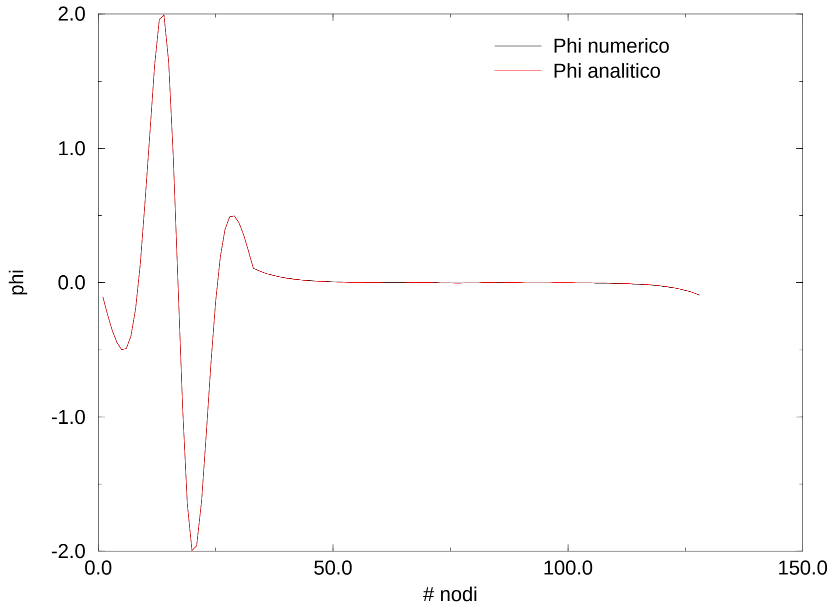

Figura 5.15: Potenziale al contorno nel caso di condizioni al contorno Dirichlet e Neumann: caso del dominio rettangolare, $\varphi = \exp y \sin x$, numero di raffinamenti=5.

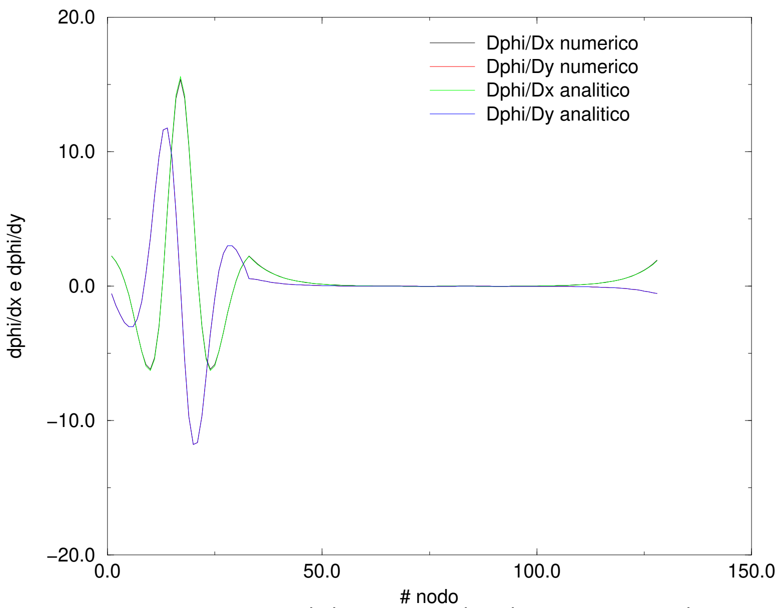

Figura 5.16: Derivate del potenziale al contorno nel caso di condizioni al contorno Dirichlet e Neumann: caso del dominio rettangolare, $\varphi = \exp y \sin x$, numero di raffinamenti=5.

Capitolo 6

Evoluzione nel tempo

Introduzione Allo scopo di verificare l'attendibilità delle stime delle derivate del potenziale ottenute tramite l'algoritmo di "patch recovery", si è affrontato il problema dell'evoluzione temporale della superficie libera di un liquido all'interno di un contenitore a cielo aperto. Al momento ci si è limitati a considerare il caso di una *standing wave* ($x = A * \cos \frac{2\pi * x}{L}$.
Il metodo scelto per l'integrazione nel tempo è un metodo di tipo *predictor-corrector*, più efficiente del metodo *Runge-Kutta* dello stesso ordine, dato il basso costo in termini di calcolo ([15]). Tale vantaggio però persiste solo se si procede a passo costante.
Prima di procedere all'integrazione nel tempo, era necessario implementare un algoritmo per l'aggiornamento della mesh tra un istante ed il successivo. L'algoritmo prescelto prevede la regolarizzazione dello spostamento dei nodi di bordo mediante la risoluzione di un problema ellittico ([12]).

Aggiornamento della mesh In letteratura sono state proposte moltissime strategie per l'aggiornamento della mesh tra un istante di tempo ed il successivo. Alcune di queste sono citate in [12]. Quella qui adottata "consiste nel regolarizzare lo spostamento dei nodi di bordo (noto) mediante la risoluzione di un problema ellittico" ([12]).
Indichiamo con $\Omega(t^n)$ il dominio all'istante t^n e sia $\eta(\mathbf{x}^n, t^{n+1})$ lo spostamento noto all'istante t^{n+1} dei nodi di bordo della mesh ($\mathbf{x}^n \in \partial\Omega(t^n)$). Si vuole determinare la nuova posizione dei nodi interni, compatibile con lo spostamento dei nodi di bordo e tale da mantenere inalterate le connessioni. La nuova posizione viene determinata nel modo seguente:

$$\mathbf{x}^{n+1} = \mathbf{x}^n + \mathbf{s}^{n+1}(\mathbf{x}^n) \tag{6.1}$$

dove lo spostamento $\mathbf{s}^{n+1}$ si calcola risolvendo il sistema:

$$\left\{ \begin{array}{lcll} \nabla^2 \mathbf{s}^{n+1} & = & 0 & \text{in} \quad \Omega(t^n) \\ \mathbf{s}^{n+1} & = & \eta(t^{n+1}) & \text{su} \quad \Gamma(t^n) \end{array} \right. \tag{6.2}$$

ovvero in forma debole:
Trovare $\mathbf{s}^{n+1} \in \mathbf{H}^1(\Omega(t^n))$ *tale che:*

$$\left\{ \begin{array}{lcll} (\nabla \mathbf{s}^{n+1}, \nabla \mathbf{v}) & = & 0 & \forall \mathbf{v} \in \mathbf{H}^1_0(\Omega(t^n)) \\ \mathbf{s}^{n+1} & = & \eta & \text{su} \quad \Gamma(t^n) \end{array} \right. \tag{6.3}$$

Risolvendo l'equazione di Laplace, si ha l'effetto di regolarizzare gli spostamenti e pertanto la nuova griglia è generalmente ancora accettabile. Tuttavia, dopo un certo numero di passi temporali, la griglia può risultare eccessivamente distorta e diventa necessario ri-triangolare il dominio e interpolare la soluzione sulla nuova griglia. Nel nostro caso, tuttavia, non essendosi terminato l'algoritmo di avanzamento nel tempo, non si è potuto stabilire se si verifichi questa eventualità.
Il sistema è stato risolto con un metodo ad elementi finiti. Inoltre, siccome le singole componenti dello spostamento sono disaccoppiate, è stato risolto separatamente un problema di Laplace per la componente x ed uno per la componente y.

Metodo Adams-Bashfort-Moulton Tra i metodi predictor-corrector i più popolari sono gli schemi detti *Adams-Bashfort-Moulton*, che godono di buone proprietà di stabilità; tra questi è stato scelto un algoritmo del quarto ordine.
Supponiamo si debba risolvere un'equazione differenziale del primo ordine:

$$\frac{dy}{dx} = f(x, y) \tag{6.4}$$

l'integrazione avviene attraverso quattro passi successivi (in effetti il quarto è opzionale):

- *predictor:*

$$y_1 = y_0 + \frac{h}{24}[55 f_0 - 59 f_{-1} + 37 f_{-2} - 9 f_{-3}] + O(h^5) \tag{6.5}$$

- *evaluation of the derivative:*

$$f_1 = f(x, y_1) \tag{6.6}$$

- *corrector:*

$$y_1 = y_0 + \frac{h}{24}[9f_1 + 19f_0 - 5f_{-1} + f_{-2}] + O(h^5) \tag{6.7}$$

- *evaluation 2:*

$$f_1 = f(x, y_1) \tag{6.8}$$

Nel caso specifico le equazioni differenziali ordinarie che devono eseere integrate sono due:

- Condizioni dinamica per la superficie libera (linearizzata):

$$\frac{D\phi}{Dt} = -g\eta \tag{6.9}$$

- evoluzione della superficie libera

$$\frac{D\mathbf{x}}{Dt} = \nabla\phi \tag{6.10}$$

Descrizione dell'algoritmo

Condizioni iniziali Per quanto riguarda l'avanzamento in tempo, per poter applicare l'algoritmo descritto è necessario conoscere i valori di $-g\eta$ e di $\nabla\varphi$, ovvero della velocità, per l'istante zero e per tre istanti successivi. Nel nostro caso, si ha a che fare con un'onda stazionaria di equazione:

$$\eta = A * \cos\frac{2\pi}{L}x \tag{6.11}$$

Pertanto si dovrebbero conoscere i valori di f_0, f_1, f_2, f_3 e di $\mathbf{v}_0$, $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$ sulla superficie libera. Per quanto riguarda l'istante zero, sono note le condizioni iniziali e pertanto si può porre:

$$f_0 = -g\eta(x_0) \tag{6.12}$$

$$\mathbf{v}_0 = 0 \tag{6.13}$$

Per ricavare i corrispondenti valori agli istanti 1, 2 e 3 è consigliato di ricorrere ad altri metodi di integrazione a passo unico, quali ad esempio il *Metodo di Runge-Kutta* ([13], [9]). In particolare si è utilizzato il metodo Runge-Kutta del quarto ordine descritto in 2.1.

Ciclo nel tempo Fissato un intervallo temporale dt e il numero di passi previsti, per ogni passo t si applicherà il procedimento prima descritto. Per $t = 4, nstep$

- Predictor

$$
\begin{aligned}
x_t &= x_{t-1} + \frac{dt}{24}[55u_{t-1} - 59u_{t-2} + 37u_{t-3} - 9u_{t-4}] && (6.14)\\
y_t &= y_{t-1} + \frac{dt}{24}[55v_{t-1} - 59v_{t-2} + 37v_{t-3} - 9v_{t-4}] && (6.15)\\
\phi_t &= \phi_{t-1} + \frac{dt}{24}[55f_{t-1} - 59f_{t-2} + 37f_{t-3} - 9f_{t-4}] && (6.16)
\end{aligned}
$$

- Evaluation: si risolve sull'intero dominio il problema ellittico

$$\nabla^2\phi = 0 \tag{6.17}$$

con le seguenti condizioni al contorno:

  - Sulla superficie libera si impongono i valori di ϕ appena calcolati
  - Sulle pareti la $\frac{\partial\phi}{\partial n}$ deve essere uguale a zero

Il dominio è quello delimitato dalle pareti del contenitore e dalla superficie libera come valutata al passo predictor.
In questo modo, si trovano i valori di ϕ e delle sue derivate (e quindi della velocità) su tutti i nodi della griglia.

- Corrector:

$$
\begin{aligned}
x_t &= x_{t-1} + \frac{dt}{24}[9u_t + 19u_{t-1} - 5u_{t-2} + u_{t-3}] && (6.18)\\
y_t &= y_{t-1} + \frac{dt}{24}[9v_t + 19v_{t-1} - 5v_{t-2} + v_{t-3}] && (6.19)\\
\phi_t &= \phi_{t-1} + \frac{dt}{24}[9f_t + 19f_{t-1} - 5f_{t-2} + f_{t-3}] && (6.20)
\end{aligned}
$$

Si noti che:

$$f_t = -gy_t \tag{6.21}$$

- Evaluation 2: si potrebbe ripetere il passo evaluation riaggiornando il dominio e imponendo i nuovi valori di ϕ.

- Aggiornamento dei valori iniziali: Indicate con g tutte le grandezze f,u,v si pone:

$$g_{t-4} = g_{t-3} \tag{6.22}$$
$$g_{t-3} = g_{t-2} \tag{6.23}$$
$$g_{t-2} = g_{t-1} \tag{6.24}$$
$$g_{t-1} = g_t \tag{6.25}$$

Si noti che ogni volta che si modificano le coordinate dei nodi sulla superficie libera è necessario aggiornare la mesh.

Capitolo 7

Conclusioni

In questo progetto di tesi ci si proponeva di studiare il comportamento idroelastico delle strutture galleggianti.
Nell'ambito di questo progetto, i primi tre mesi di lavoro sono stati dedicati allo studio dello *sloshing* in contenitori rettangolari. Sebbene tale argomento non sia immediatamente collegato al tema principale dello stage, esso è stato scelto come caso di studio per approfondire i problemi relativi al trattamento di moti a superficie libera utilizzando diversi modelli (viscoso e inviscido-irrotazionale) e diversi metodi numerici (Elementi Finiti e Integrali di Contorno). Questo ha consentito di raggiungere una buona padronanza di tali strumenti che è risultata poi di enorme importanza nella prosecuzione del lavoro. Particolarmente proficuo è stato il confronto diretto con il lavoro dell'Ing. G. Colicchio. I casi di sloshing studiati sono stati essenzialmente casi di oscillazione orizzontale, in quanto, per motivi di tempo, non è stato possibile analizzare alcuni casi di oscillazione verticale e *standing waves*. Particolarmente importante, sarebbe anche prendere in considerazione casi di rollio, condizione particolarmente sfavorevole per quanto riguarda lo sloshing in navi adibite al trasporto di fluidi.
La realizzazione del codice per lo studio del comportamento idroelastico dei contenitori galleggianti è, invece, un compito complesso che avrebbe richiesto più tempo dei quattro mesi ad esso dedicati. L'idea di adottare diversi modelli e metodi numerici per le diverse porzioni di fluido ha reso molto articolato il lavoro da svolgere. In questa cornice, sono stati già risolti svariati problemi: lo sviluppo del codice BEM, la scelta del metodo Dirichlet-Neumann come algoritmo di accoppiamento, lo sviluppo del codice FEM e il calcolo delle derivate parziali del potenziale al contorno. Si è anche iniziato il trattamento dell'integrazione nel tempo e l'implementazione del segmento di codice per l'aggiornamento della griglia di calcolo al variare del tempo. Il limite essenziale di questa trattazione è rappresentato dal fatto che, per po-

tersi concentrare maggiormente sugli aspetti dell'accoppiamento, il modello adottato è stato in ogni caso quello di fluido non viscoso, incomprimibile in moto irrotazionale. In uno sviluppo futuro, sarà pertanto necessario, oltre che completare le parti di codice parzialmente sviluppate, introdurre un solutore per le equazioni di Navier-Stokes nonchè occuparsi della modellazione del corpo galleggiante deformabile.

Bibliografia

[1] H. Blum, J. Harig, S. Müller, P. Schreiber, e S. Turek. *Finite Element Analysis Tools, User Manual - Release 1.3*, 1995.

[2] A.G.L. Borthwick, M.S. Turnbull, e R. Eatock Taylor. Nonlinear Wave Loading Using Sigma-Transformed and Unstructured Finite Element Meshing. 2000.

[3] X. Cai, H.P. Langtangen, B.F. Nielsen, e A. Tveito. A Finite Elemt Method for Fully Nonlinear Water Waves. *Journal of Computational Physics*, (143):544–568, 1998.

[4] M.J. Chern, A.G.L. Borthwick, e R. Eatock Taylor. A Pseudospectral σ-Transformation Model of 2-D Nonlinear Waves. *Journal of Fluids and Structures*, (13):607–630, 1999.

[5] G. Colicchio. *Dinamica di Veicoli Marini con Liquidi a Bordo*. Tesi di laurea, Facoltà di Ingegneria, Università di Roma La Sapienza, 1998.

[6] A.J. Davies. *The Finite Element Method: A First Approach*, capitolo 4, pagine 105–175.

[7] P.M. Gresho e R.L. Sani. *Incompressible Flow and the Finite Element Method*, capitolo 4, pagine 847–872. 1998.

[8] P.C.A. De Haas e P.J. Zandbergen. The Application of Domain Decomposition to Time-Domain Computations of Nonlinear Water Waves with a Panel Method. *Journal of Computational Physics*, (129):332–344, 1996.

[9] N.V. Kopchenova e I.A. Maron. *Computational Mathematics - Worked Examples and Problems with Elements of Theory*, capitolo 8, pagine 231–240. 1984.

[10] I. Lodes. Simulazione Numerica di Problemi di Sloshing con Metodi ad Elementi Finiti in Domini Bidimensionali. Rapporto Tecnico 86, INSEAN, 1999.

[11] I. Lodes, G. Colicchio, G. Graziani, e G. Fotia. Confronto di Solutori FEM e Solutori BEM per lo Sloshing in Contenitori. Rapporto Tecnico 91, INSEAN, 1999.

[12] F. Nobile. *Problemi di Interazione Fluido-Struttura in Emodinamica.* Tesi di dottorato, Facoltà di Ingegneria, Politecnico di Milano, 1998.

[13] W.H. Press, B.P. Flannery, S.A. Teukolsky, e W.T.Vetterling. *NUMERICAL RECIPES- The Art of Scientific Computing*, capitolo 15, pagine 547–577. 1989.

[14] A. Quarteroni e A. Valli. *Domain Decomposition Methods for Partial Differential Equations*, capitolo 1, pagine 1–40. Clarendon Press, 1999.

[15] J. Stoer e R. Bulirsch. *Introduzione all'Analisi Numerica 2*, capitolo 7, pagine 125–128,158–160. 1975.

[16] G.X. Wu, Q.W. Ma, e R. Eatock Taylor. Numerical Simulation of Sloshing Waves in a 3D Tank Based on a Finite Element Method. Prestampa, 1998.

[17] J.Z. Zhu. A Posteriori Error Estimation-The Relationship between Different Procedures. *Computer Methods in Applied Mechanics and Engineering*, (150):411–422, 1997.

[18] O.C. Zienkiewicz e J.Z. Zhu. The Superconvergent Patch Recovery and a Posteriori Error Estimates. Part 1: The Recovery Technique. *International Journal for Numerical Methods in Engineering*, 33:1331–1364, 1992.

[19] O.C. Zienkiewicz e J.Z. Zhu. The Superconvergent Patch Recovery and a Posteriori Error Estimates. Part 2: Error Estimates and Adaptivity. *International Journal for Numerical Methods in Engineering*, 33:1365–1382, 1992.