UNIVERSITÀ DEGLI STUDI DI ROMA
"LA SAPIENZA"

Facoltà di Scienze Matematiche, Fisiche e Naturali

Tesi di Laurea in Matematica

ALGORITMI VELOCI PER L'EVOLUZIONE DEI FRONTI

Studente:
Emiliano Cristiani

Relatore:
Prof. Maurizio Falcone

a.a. 2001-2002

Everything should be made as simple

as possible, but not simpler.

Albert Einstein

Indice

Introduzione

Lo scopo di questa tesi è lo studio di alcuni metodi di approssimazione per l'equazione

$$\begin{cases} c(x)|\nabla T(x)| = 1 & x \in \Omega \\ T(x) = 0 & x \in \partial\Omega \end{cases} \tag{1}$$

dove $c(x)$ è una funzione nota e Ω un aperto di $\mathbb{R}^n$. Questa equazione, insieme alla sua versione evolutiva,

$$\begin{cases} u_t + c(x)|\nabla u(x,t)| = 0 & x \in \Omega\,,\ t > 0 \\ u(x,0) = u_0(x) & x \in \Omega \end{cases} \tag{2}$$

è stata oggetto di numerose ricerche negli ultimi dieci anni per la sua importanza in vari campi applicativi. Ad esempio, queste equazioni compaiono nello studio dei problemi di tempo minimo della teoria del controllo ottimo, nello studio della evoluzione dei fronti con applicazioni alla combustione ed alla crescita di cristalli, nel trattamento delle immagini e nel calcolo delle geodetiche.
Questo ha fatto delle equazioni (1) e (2) i problemi modello nello studio delle equazioni di tipo Hamilton-Jacobi del primo ordine. I metodi analitici e numerici sviluppati per queste equazioni possono essere estesi anche a problemi non lineari più complessi del tipo

$$\begin{cases} u_t + H(x,u,\nabla u) = 0 & x \in \Omega\,,\ t > 0 \\ u(x,0) = u_0(x) & x \in \Omega \end{cases} \tag{3}$$

Alla base dei metodi analitici e numerici sviluppati per queste equazioni ci sono gli sviluppi recenti della teoria delle soluzioni di *viscosità*, introdotte da M. G. Crandall e P.-L. Lions in [8] nel 1983. In particolare, questa teoria ha permesso di ottenere risultati di esistenza e unicità nello spazio delle funzioni continue e, in tempi più recenti, anche per soluzioni discontinue sotto ipotesi abbastanza generali [28].
Per quanto riguarda la risoluzione numerica sono stati introdotti, a metà degli anni 80, alcuni metodi alle differenze finite di tipo monotòno che garantiscono la convergenza all'unica soluzione di *viscosità* [10]. Questi metodi

sono stati affiancati, a partire dalla fine degli anni 80, da schemi numerici semilagrangiani [16] che nascono, in ambito controllistico, dall'applicazione della versione discreta del Principio di Programmazione Dinamica (Bellman, 1960). Questi schemi numerici sono in generale più accurati dei precedenti e nel caso stazionario si traducono in un problema di punto fisso. È interessante osservare che i metodi alle differenze finite e quelli semilagrangiani hanno un costo computazionale simile che, in dimensione alta, può divenire proibitivo e richiedere un'implementazione su macchine parallele [15].
Per quanto riguarda l'equazione (1), nel 1996 J. A. Sethian ha sviluppato alcune idee presenti in un lavoro di J. N. Tsitsiklis introducendo i *Fast Marching Methods* (FMM). Questi metodi numerici sono basati su una discretizzazione alle differenze finite ma hanno un costo computazionale notevolmente inferiore ai metodi precedenti, dell'ordine di $O(N \ln(N))$ se N è il numero dei nodi scelti per l'approssimazione. Essi sono costruiti in modo tale da calcolare ogni nodo non più di quattro volte, mentre in un metodo iterativo classico questo può avvenire infinite volte.
Sebbene J. A. Sethian dal 1996 ad oggi abbia applicato questi metodi in numerosissimi campi, essi mancano ancora di una solida base teorica. In particolare, mancano le ipotesi sulla funzione $c(x)$ che garantiscano la validità del metodo ed un teorema che assicuri che la soluzione calcolata da questi metodi è identica a quella calcolata con la procedura di punto fisso. L'obiettivo principale di questa tesi è quello di chiarire dal punto di vista matematico le proprietà del FMM dando una dimostrazione della convergenza. Nella parte finale della tesi verranno anche introdotte alcune generalizzazioni che si sono dimostrate particolarmente veloci ed efficaci nell'approssimazione di alcuni problemi modello.

La struttura della tesi è la seguente.
Nel primo capitolo vengono studiati i principali risultati teorici noti per l'equazione (1), introducendo in particolare la nozione di soluzione di *viscosità*. Si descrive il problema dell'evoluzione dei fronti ed il problema di tempo minimo, entrambi riconducibili alla risoluzione dell'equazione (1). Vengono poi presentate le idee di base di due schemi numerici "classici" basati sulla tecnica di punto fisso per i quali esiste una dimostrazione della convergenza alla soluzione di *viscosità* e numerosi altri risultati teorici.
Nel secondo capitolo viene studiato approfonditamente il FMM. In particolare, viene data una dimostrazione rigorosa della sua validità, introducendo una condizione che lega il passo di discretizzazione Δx e la funzione $c(x)$. Questa ipotesi è assente nei lavori di J. A. Sethian. Diamo poi la dimostrazione della convergenza del FMM alla soluzione di *viscosità* riconducendoci allo schema alle differenze finite basato sulla tecnica di punto fisso.

Nel terzo capitolo viene formulato un nuovo schema numerico, che nasce dalla fusione del FMM con lo schema semilagrangiano con l'obiettivo di ereditare i pregi dell'uno e dell'altro metodo. Ne abbiamo dimostrato la validità, indipendentemente dalla condizione sul passo Δx introdotta nel FMM.
Nel quarto capitolo proponiamo alcune varianti per migliorare ulteriormente il metodo descritto nel terzo capitolo. La prima consiste nell'evitare di ricalcolare il valore T in un nodo se esso sarà identico al valore precedentemente assegnatogli: questo risultato viene ottenuto introducendo una condizione che permette di decidere *a priori* se il calcolo che si sta per effettuare porterà o meno ad un valore diverso per il nodo in questione. Il secondo raffinamento consiste in una diversa gestione dell'ordine con cui vengono calcolati i nodi della griglia e permette di trattare anche casi più generali dell'equazione (1). Esso è stato implementato solamente nel caso particolare in cui $c(x) = c_0$, nel quale si raggiunge il minimo costo computazionale teoricamente possibile, cioè $O(N)$.

Infine, nell'ultimo capitolo, abbiamo sviluppato numerosi test per confrontare le prestazioni dei metodi studiati e abbiamo poi passato in rassegna le più note applicazioni dell'equazione (1): evoluzione dei fronti, ricerca di traiettorie che minimizzano il tempo di percorrenza nel piano e su superfici, Shape from Shading e una interessante applicazione al modello di Poincaré per la geometria iperbolica.

Capitolo 1

Teoria dell'evoluzione dei fronti e problemi di controllo ottimo

1.1 Teoria dell'evoluzione dei fronti

Consideriamo una curva chiusa

$$\begin{array}{rcll} \Gamma_0 : [0,S] & \to & \mathbb{R}^n & \qquad x(0) = x(S) \\ s & \mapsto & x(s) & \end{array}$$

e supponiamo che ogni punto della curva si muova sotto l'azione di un campo di velocità $\mathbf{c}$, considerato noto.
Il nostro scopo è quello di prevedere l'evoluzione della curva (il fronte) sotto l'azione di questo campo del velocità.
Il campo di velocità $\mathbf{c}$ può dipendere da molti fattori, tra cui, ad esempio, il tempo, la posizione, la curvatura, il vettore normale alla curva nel punto, etc. Noi ci occuperemo del caso in cui

$$\mathbf{c}(x) = c(x)\widehat{n}_{ext}(x)\,, \qquad c : \mathbb{R}^n \to \mathbb{R} \tag{1.1}$$

cioè supporremo che ogni punto della curva si sposti con una velocità pari a $c(x)$ nella direzione della normale esterna alla curva in quel punto. Da ora in poi indicheremo la normale esterna semplicemente con n. Inoltre supporremo

$$c \in Lip(\mathbb{R}^n) \cap L^\infty(\mathbb{R}^n).$$

Consideriamo ora la traiettoria che ogni punto della curva traccia sotto l'azione del campo di velocità e parametrizziamola con il parametro t (vedi Figura 1.1). Per ogni $\bar{s} \in [0,S]$ *fissato*, il punto $x(\bar{s})$ della curva iniziale percorre nel piano la traiettoria

$$x(\bar{s},t) \qquad t \in [0,+\infty).$$

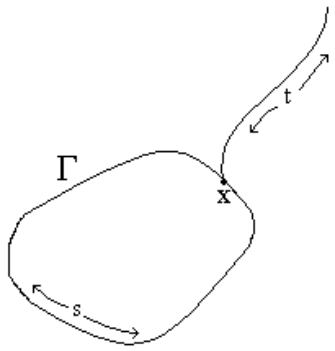


Figura 1.1: curva tracciata da un singolo punto del fronte

Si ottiene così la famiglia ad un parametro di curve

$$\Gamma_t = \Big\{ x(s,t)\,,\ s \in [0,S] \Big\}\,, \qquad \text{al variare di } t \in [0,+\infty).$$

La curva Γ_t rappresenta quindi l'evoluzione del fronte iniziale Γ_0 al tempo t. La velocità di ogni punto della curva è data dal vettore $x_t(s,t)$, indicando con x_t la derivata parziale rispetto al tempo di x. Dalla (1.1), si ha:

$$x_t = \mathbf{c}(x) = c(x)n.$$

Esistono vari modi per ricavare un'equazione la cui soluzione sia l'evoluzione della curva ad ogni istante (si veda, ad esempio, [24]) e, nei casi più semplici, è anche possibile trovare una soluzione analitica esplicita. D'altra parte, anticipiamo già ora che le ipotesi

$$\Gamma_0 \in C^\infty([0,S]) \qquad \text{e} \qquad c \in C^\infty(\mathbb{R}^n) \cap L^\infty(\mathbb{R}^n)$$

non assicurano che $\Gamma_t \in C^1$ per ogni $t \in [0,+\infty)$, cioè non si può escludere la possibilità che ad un certo istante si formino degli spigoli nel fronte. La formazione di uno spigolo fa perdere la possibilità di definire la normale in ogni punto della curva, aprendo la strada a diverse interpretazioni possibili di cosa sia l'evoluzione del fronte da quel momento in poi. Ad esempio, se continuiamo a tracciare la traiettoria seguita da ogni punto, assistiamo all'autointersezione del fronte. Una situazione di questo tipo è illustrata nella Figura 1.2, dove possiamo vedere l'evoluzione della curva a quattro istanti successivi, $t_0 = 0 < t_1 < t_2 < t_3$. Come è facile vedere, l'evoluzione

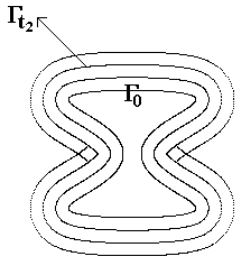


Figura 1.2: autointersezione del fronte

nella direzione normale ha fatto sì che Γ_{t_2} abbia formato uno spigolo, e che successivamente il fronte si sia autointersecato. Poiché il fronte rappresenta, nelle più comuni applicazioni fisiche, l'interfaccia tra due regioni di spazio, è chiaro che l'autointersezione del fronte sia - dal punto di vista fisico - un fenomeno privo di senso.
Un classico esempio è dato dalla propagazione di un fronte di fiamma, in cui si vuole prevedere l'evoluzione della zona di confine tra la zona bruciata e quella ancora non bruciata. In particolare, si può pensare ad un gas che brucia all'interno di un contenitore (vedi Figura 1.3) o ad un incendio in una foresta.

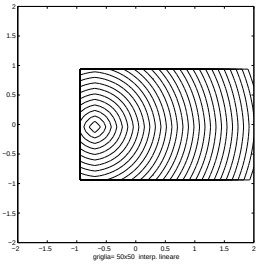


Figura 1.3: fronte di fiamma che si espande in un contenitore

Nel paragrafo 1.4 riprenderemo in esame il problema della perdita di regolarità del fronte.

Il metodo *level set*

Il metodo *level set* è una delle più valide tecniche per ricavare l'equazione del moto dei fronti. Esso è stato introdotto all'inizio degli anni 80 da S. Osher e J. A. Sethian ed è stato in seguito sviluppato con la collaborazione di altri

autori ed applicato in situazioni più generali. Rimandiamo a [24] e [17] per uno studio approfondito di questa tecnica.
L'idea di base è quella di pensare il fronte Γ_0 come la curva di livello relativa al valore 0 di una funzione di n variabili (il problema quindi "cresce" di una dimensione). Il nostro obiettivo diventa quello di trovare una funzione $u(x,t)$ tale che, per ogni t fissato, la sua curva di livello relativa al valore 0 sia esattamente la curva al tempo t, cioè

$$\Gamma_t = \Big\{ x \in \mathbb{R}^n \ : \ u(x,t) = 0 \Big\}, \quad t \in [0, +\infty).$$

Inoltre, imponiamo che la curva di livello relativa al valore 0 della funzione $u(x,0)$ sia proprio la curva iniziale Γ_0, cioè

$$\Gamma_0 = \Big\{ x \in \mathbb{R}^n \ : \ u(x,0) = 0 \Big\}. \tag{1.2}$$

I dati del problema sono il campo $\mathbf{c}$ e la curva iniziale Γ_0.

Per ricavare un'equazione di cui u sia soluzione, procediamo nel seguente modo: poniamo

$$u(x,t) = \pm d(x, \Gamma_t)$$

dove d è la distanza tra il punto x e la curva Γ_t, con il segno $+$ se il punto è esterno alla curva e con il segno $-$ se è interno.
Notiamo che con questa definizione la condizione (1.2) è automaticamente soddisfatta.
Sia $x(t)$ la traiettoria che un generico punto (fissato) della curva traccia sul piano sotto l'azione di $\mathbf{c}$. Dobbiamo quindi imporre che sia

$$u(x(t),t) = 0 \qquad \forall t \in [0, +\infty)$$

cioè che al tempo t il punto $x(t)$ abbia distanza *nulla* dalla curva (sia cioè un punto della curva).
Derivando rispetto al tempo quest'ultima equazione, e indicando con $\nabla u = (u_{x_1}, \ldots, u_{x_n})$, otteniamo

$$u_t + \nabla u(x(t),t) \cdot x_t(t) = 0 \tag{1.3}$$

È noto che, *per ogni t fissato*, la normale n alla curva di livello relativa al valore 0 di $u(x,t)$ nel punto $x(t)$ (che appartiene a questa curva di livello) è data da

$$n(x(t)) = \frac{\nabla u(x(t),t)}{|\nabla u(x(t),t)|}$$

ed inoltre essa è la normale *esterna*, perché per definizione la u è crescente verso l'esterno della curva.
Inoltre sappiamo per ipotesi che $x_t(t) = c(x)n$. Quindi, sostituendo nella (1.3), otteniamo

$$u_t + \nabla u \cdot x_t = u_t + \nabla u \cdot c(x)n = \tag{1.4}$$
$$= u_t + c(x)\nabla u \cdot \frac{\nabla u}{|\nabla u|} = u_t + c(x)|\nabla u| = 0$$

In conclusione, l'equazione da risolvere è

$$\begin{cases} u_t + c(x)|\nabla u| = 0 & x \in \mathbb{R}^n,\ t > 0 \\ u(x, t = 0) = \pm d(x, \Gamma_0) & x \in \mathbb{R}^n \end{cases} \tag{1.5}$$

L'equazione appena ricavata è un caso particolare della ben nota equazione di Hamilton-Jacobi evolutiva

$$\begin{cases} u_t + H(x, u, \nabla u) = 0 & x \in \mathbb{R}^n,\ t > 0 \\ u(x, t = 0) = u_0(x) & x \in \mathbb{R}^n \end{cases}$$

dove $H : \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}$, per la quale esistono numerosi risultati teorici e numerici, grazie anche al suo stretto legame con le leggi di conservazione. Per una trattazione completa si veda [11], [3] e [21].

Concentriamo ora la nostra attenzione al caso in cui il fronte abbia un'evoluzione strettamente monotòna, cioè

$$\forall\ t_1, t_2 > 0 \qquad t_1 > t_2 \Rightarrow \Gamma_{t_1} \supset \Gamma_{t_2} \qquad \text{(fronte in espansione)}$$

oppure

$$\forall\ t_1, t_2 > 0 \qquad t_1 > t_2 \Rightarrow \Gamma_{t_1} \subset \Gamma_{t_2} \qquad \text{(fronte in contrazione)}$$

Questi due tipi di evoluzione sono garantiti rispettivamente dalle condizioni:

$$c(x) > 0 \qquad \forall x \in \mathbb{R}^n$$

e

$$c(x) < 0 \qquad \forall x \in \mathbb{R}^n.$$

In questo caso l'equazione (1.4) può essere scritta in una forma semplificata, introducendo il seguente cambio di variabili

$$u(x, t) = T(x) - t.$$

Sostituendo, si ottiene facilmente

$$c(x)|\nabla T| = 1 \qquad\qquad x \in \mathbb{R}^n \tag{1.6}$$

Il fronte Γ_t è ora dato da tutte le curve di livello della funzione T, cioè

$$\Gamma_t = \left\{ x \in \mathbb{R}^n \ : \ T(x) = t \right\}, \qquad t \in [0, \infty)$$

Il vantaggio di questa formulazione è evidente: per conoscere Γ_t ad ogni istante è sufficiente trovare *una sola* funzione $z = T(x)$ e poi guardare le sue curve di livello. Anche dal punto di vista numerico, appare già da ora chiara la possibilità di risparmiare molto in termini di memoria utilizzata dal calcolatore rispetto all'equazione non stazionaria.

Nel caso $c(x) > 0$, cioè di un fronte che si espande, la regione di spazio in cui dobbiamo ricostruire la funzione T è $\mathbb{R}^n \backslash \Omega$, dove con Ω indichiamo la regione delimitata da Γ_0 (vedi Figura 1.4).

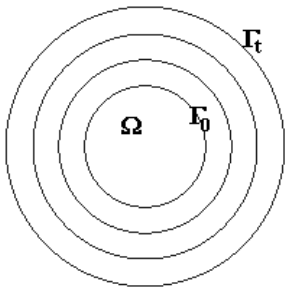


Figura 1.4: fronte che si espande (curve di livello della funzione T)

All'equazione (1.6) possiamo quindi affiancare la condizione al bordo

$$T(x) = 0 \quad \text{se} \quad x \in \Gamma_0$$

In conclusione, possiamo riscrivere l'equazione (1.5) nel modo seguente:

$$\begin{cases} c(x)|\nabla T| = 1 & x \in \mathbb{R}^n \backslash \Omega \\ T(x) = 0 & x \in \Gamma_0 = \partial\Omega \end{cases} \tag{1.7}$$

Nel paragrafo 1.4 enunceremo un teorema "ponte" che lega in modo rigoroso le equazioni (1.5) e (1.7).

Per quanto riguarda il comportamento della funzione T se $|x| \to +\infty$, è naturale la seguente condizione:

$$\lim_{|x|\to+\infty} T(x) = +\infty.$$

Come vedremo, gli schemi numerici che approssimano l'esatta soluzione della (1.7), soddisfano automaticamente questa condizione.

Nel caso in cui $c(x) < 0$, invece, il fronte evolverà all'interno di Ω. L'equazione da risolvere sarà quindi:

$$\begin{cases} c(x)|\nabla T| = 1 & x \in \Omega \\ T(x) = 0 & x \in \Gamma_0 = \partial\Omega \end{cases}$$

Concludiamo questo paragrafo con due osservazioni:

1. Il metodo *level set* permette di trattare anche il caso in cui Ω è sconnesso. Si veda ad esempio la Figura 1.5 in cui il fronte iniziale è composto da due circonferenze separate. Esse, evolvendo con velocità positiva, ad un certo istante si toccano e si fondono insieme. Confrontando le Figure 1.5 e 1.2, possiamo già da ora osservare che il metodo *level set* risolve il problema dell'autointersezione del fronte. Infatti, vedremo che le curve di livello della funzione T hanno la proprietà di essere strettamente contenute l'una nell'altra.

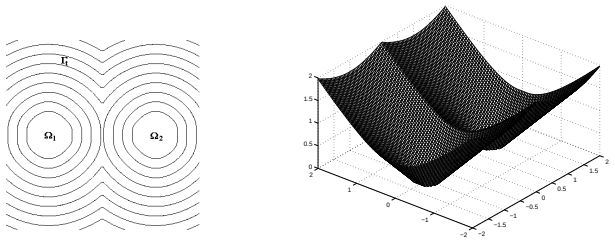


Figura 1.5: Ω sconnesso (funzione T e sue curve di livello)

2. Se l'evoluzione del fronte non fosse monotòna, la formulazione stazionaria del problema non sarebbe ammissibile. Infatti, in questo caso, il fronte potrebbe attraversare per più di una volta lo stesso punto del dominio e questo richiederebbe che la funzione T possa assumere più valori in un stesso punto.

1.2 Problemi di controllo ottimo

Gli elementi principali che costituiscono un problema di controllo ottimo sono la *dinamica* e il *funzionale costo*.
La *dinamica* è un sistema di equazioni differenziali ordinarie *controllato*, nel senso che ora specificheremo. Esso si può scrivere nella forma

$$\begin{cases} \dot{y}(t) = b(y(t), \alpha(t)) & t \in I \\ y(0) = x \end{cases} \tag{1.8}$$

dove

- I è un intervallo (temporale) di $\mathbb{R}^+$
- $\alpha(\cdot) \in \mathcal{A} = \left\{ z(\cdot) : I \to A \mid A \subset \mathbb{R}^m,\ A \text{ compatto},\ z \in L^\infty(I) \right\}$
- $b : \mathbb{R}^n \times A \to \mathbb{R}^n$
- $x \in \mathbb{R}^n$

e la curva $y : I \to \mathbb{R}^n$ è la soluzione del sistema. Essa viene solitamente indicata con $y(t; \alpha)$ per mettere in evidenza la dipendenza dalla funzione α. Supponiamo per il momento di conoscere b, α e x. In questo caso la *dinamica* diventa un consueto sistema di equazioni differenziali ordinarie. L'ipotesi

$$b(y, \alpha) \in C^0(\mathbb{R}^n \times A) \cap L^\infty(\mathbb{R}^n \times A) \text{ } \textit{e lipschitziana rispetto a } y \tag{1.9}$$

ci assicura esistenza e unicità della soluzione per $t \in I$. Ora invertiamo il punto di vista: la funzione $\alpha(t)$ (detta *controllo*) diventa la vera incognita del problema. Lo scopo è ora quello di trovare un controllo α che faccia soddisfare alla soluzione del sistema $y(t)$ alcune condizioni imposte *a priori*. In particolare, viene richiesto che la soluzione $y(t)$ sia la curva di "costo minimo" tra tutte le possibile curve soluzioni di (1.8) al variare di $\alpha \in \mathcal{A}$

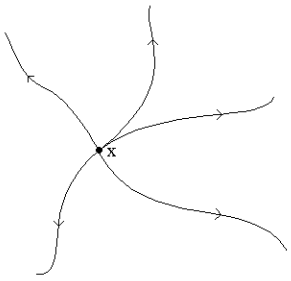


Figura 1.6: possibili traiettorie $y(t)$ al variare di $\alpha \in \mathcal{A}$

(vedi Figura 1.6). Il "costo" di una traiettoria viene definito attraverso un funzionale J_x, detto appunto *funzionale costo*, che associa ad ogni controllo (e quindi alla curva $y(t;\alpha)$) un numero reale. Osserviamo che il pedice nel simbolo J tiene memoria del dato iniziale.
Il problema si riduce quindi a trovare, se esiste, un controllo $\alpha^*(t)$ che realizzi il minimo di questo funzionale. Esso sarà detto *controllo ottimo*.
Definiamo infine la *funzione valore* $u(x)$ nel seguente modo:

$$u(x) = \inf_{\alpha(\cdot)\in\mathcal{A}} J_x(\alpha(\cdot))$$

Essa associa ad ogni dato iniziale x del sistema (1.8) il costo della traiettoria corrispondente al controllo ottimo.

A titolo di esempio mostriamo il *funzionale costo* legato ai cosiddetti problemi a *orizzonte infinito*. In questo caso si ha $I = [0, +\infty]$ e il funzionale è

$$J_x(\alpha(\cdot)) = \int_0^{+\infty} f(y(s;\alpha), \alpha(s))e^{-\lambda s}ds$$

dove $\lambda \in \mathbb{R}^+$ e

$$f : \mathbb{R}^n \times A \to \mathbb{R}$$

La funzione f è detta *costo di percorrenza*. Grazie al Principio della Programmazione Dinamica (PPD), introdotto da Bellman nel 1960, è possibile dimostrare, sotto opportune ipotesi per f, che la *funzione valore* $u(x)$ è soluzione della seguente equazione

$$\lambda u(x) + \max_{a\in A}\Big\{ - b(x,a)\cdot\nabla u(x) + f(x,a)\Big\}, \quad x \in \mathbb{R}^n. \tag{1.10}$$

Altri esempi possono essere trovati in [1], insieme ad una trattazione completa dell'argomento. In [11] viene trattato in maniera molto chiara e approfondita il problema a *orizzonte finito*.

Problemi di tempo minimo

Noi ci occuperemo nel dettaglio solamente del cosiddetto problema di *tempo minimo*, nei quali il *funzionale costo* ha una forma particolarmente semplice. Consideriamo nuovamente il sistema (1.8) scegliendo $I = [0, \infty)$ e indicando ancora con $y(t;\alpha)$ la soluzione relativa al controllo α. Introduciamo inoltre nel problema un insieme $\mathcal{T} \subset \mathbb{R}^n$, detto *target*. Supponiamo inoltre che il punto iniziale x del sistema sia esterno a $\mathcal{T}$. Il funzionale costo è definito nel seguente modo:

$$J_x(\alpha(\cdot)) = \inf\Big\{t \in [0, +\infty) : y(t;\alpha) \in \mathcal{T}\Big\} \tag{1.11}$$

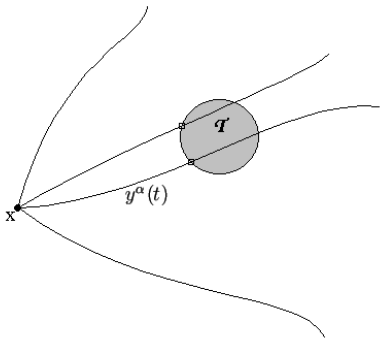


Figura 1.7: problema di *tempo minimo*

e di conseguenza la *funzione valore* sarà

$$T(x) = \inf_{\alpha(\cdot)\in\mathcal{A}} J_x(\alpha(\cdot)). \tag{1.12}$$

che in questo caso viene anche detta funzione di *tempo minimo*.
In sostanza, come si può vedere nella Figura 1.7, $J_x(\alpha(\cdot))$ è il tempo di primo arrivo della curva y a partire dal punto x con il controllo α, e $T(x)$ è il tempo di primo arrivo della curva y a partire dal punto x lungo la traiettoria più veloce possibile, relativa quindi al controllo ottimo.
Se $x \in \mathcal{T}$, allora si pone $T(x) = 0$, perché in questo caso il tempo impiegato per raggiungere il *target* è ovviamente nullo.
Inoltre, se non esiste nessun controllo che permette alla curva $y(t)$ di raggiungere il target a partire dal punto x, si pone $T(x) = +\infty$. È quindi possibile che ci siano delle regioni di $\mathbb{R}^n$ nelle quali la funzione T non è definita. Indicheremo con $\mathcal{R}$ l'insieme di definizione della funzione T. È inoltre teoricamente possibile che la funzione T sia definita solamente per $x \in \mathcal{T}$. Per evitare questo caso banale è sufficiente aggiungere delle ipotesi di *controllabilità* sul campo vettoriale b. Ad esempio si può imporre che

$$\forall x \in \partial\mathcal{T} \quad \exists \bar{a} \in A \ : \ b(x, \bar{a}) \cdot n(x) < 0$$

dove con n abbiamo indicato la normale esterna a $\mathcal{T}$. Come è dimostrato in [1], questa ipotesi garantisce che a patto di partire abbastanza vicino al *target*, esiste una traiettoria che lo raggiunge in un tempo finito.

Come nel caso "*orizzonte infinito*", il PPD permette di dimostrare che la funzione $T(x)$ è soluzione dell'equazione

$$\begin{cases} \max\limits_{a\in A}\{-b(x,a)\cdot\nabla T(x)\} = 1 & x \in \mathcal{R}\backslash\mathcal{T} \\ T(x) = 0 & x \in \mathcal{T} \end{cases}$$

Dal momento però che il dominio della funzione T non è noto *a priori* il problema diventa notevolmente più complesso, rientrando nella categoria dei problemi differenziali con *frontiera libera*.
Questa difficoltà può essere completamente superata introducendo la seguente trasformazione (monotòna), utilizzata per la prima volta da Kruzkov in [20]:

$$v(x) = 1 - e^{-T(x)}, \qquad T(x) = -\ln(1 - v(x)). \tag{1.13}$$

Come vedremo nel paragrafo 1.4 e come è ovvio dall'interpretazione fisica del problema, la funzione T assume valori nell'intervallo $[0, +\infty]$ corrispondente all'intervallo $[0, 1]$ per la funzione v.
Il PPD permette di dimostrare il seguente fondamentale

Teorema 1.1 *Supponiamo che valga l'ipotesi* (1.9) *e che $\mathcal{T}$ sia la chiusura di un aperto con frontiera regolare.*
Allora, la funzione v è soluzione di

$$\begin{cases} v(x) + \max\limits_{a \in A}\{-b(x,a) \cdot \nabla v(x)\} = 1 & x \in \mathbb{R}^n \backslash \mathcal{T} \\ v(x) = 0 & x \in \mathcal{T} \end{cases} \tag{1.14}$$

Una volta risolta questa equazione è possibile risalire al dominio $\mathcal{R}$ ponendo semplicemente

$$\mathcal{R} = \Big\{ x \in \mathbb{R}^n \ : \ v(x) < 1 \Big\}$$

che corrisponde a trovare la regione di spazio dove la funzione T assume un valore finito.

Notiamo infine che in quest'ultima equazione - così come nella (1.10) - non compare un minimo al variare di $\alpha \in \mathcal{A}$, ma più semplicemente un minimo al variare di $a \in A$. Questo permette un approccio numerico altrimenti impossibile.

1.3 Equivalenza dei due problemi

Lo scopo di questo paragrafo è di mostrare la stretta connessione esistente tra il problema dell'evoluzione dei fronti affrontata nel paragrafo 1.1 e il problema di tempo minimo descritto nel paragrafo precedente.
Per prima cosa mostriamo che i due problemi conducono alla medesima equazione per una opportuna scelta della *dinamica*. Consideriamo il sistema (1.8) ponendo

$$b(y, \alpha) = -c(y)\alpha \tag{1.15}$$

dove

$$c : \mathbb{R}^n \to \mathbb{R}\,, \qquad c \in Lip(\mathbb{R}^n) \cap L^\infty(\mathbb{R}^n)\,, \qquad c(x) > 0\,, \quad \forall x$$

e

$$\alpha(t) \in A = B(0,1) \subset \mathbb{R}^n. \tag{1.16}$$

Il significato fisico di questa scelta è particolarmente facile da descrivere: a partire da un punto x del piano, la traiettoria $y(t)$ viene percorsa alla velocità (sempre positiva) $c(y(t))$ la cui direzione varia nella palla unitaria. Il segno "$-$" nella (1.15) non ha alcuna influenza dal momento che

$$\Big\{\alpha \ : \ \alpha(t) \in B(0,1)\Big\} = \Big\{\alpha \ : \ -\alpha(t) \in B(0,1)\Big\}.$$

Di conseguenza si avrà sicuramente

$$\mathcal{R} = \mathbb{R}^n.$$

Possiamo quindi tornare senza alcun problema alla formulazione dell'equazione (1.14) in termini della funzione $T(x)$, nel seguente modo:

$$v(x) + c(x) \max_{a \in B(0,1)} \{a \cdot \nabla v(x)\} = 1 \tag{1.17}$$

Notiamo che il $\max_{a \in B(0,1)}\{a \cdot \nabla v(x)\}$ è raggiunto quando il vettore a è massimo in modulo e parallelo al vettore $\nabla v(x)$, cioè quando

$$a = a^* = \frac{\nabla v(x)}{|\nabla v(x)|} \tag{1.18}$$

Sostituendo quest'ultimo risultato nella (1.17), otteniamo

$$v(x) + c(x)|\nabla v(x)| = 1.$$

Ricordando la (1.13), otteniamo

$$1 - e^{-T(x)} + c(x)|e^{-T(x)}\nabla T(x)| = 1$$

che equivale a

$$e^{T(x)}\Big(- e^{-T(x)} + c(x)e^{-T(x)}|\nabla T(x)|\Big) = 0$$

e quindi

$$-1 + c(x)|\nabla T(x)| = 0.$$

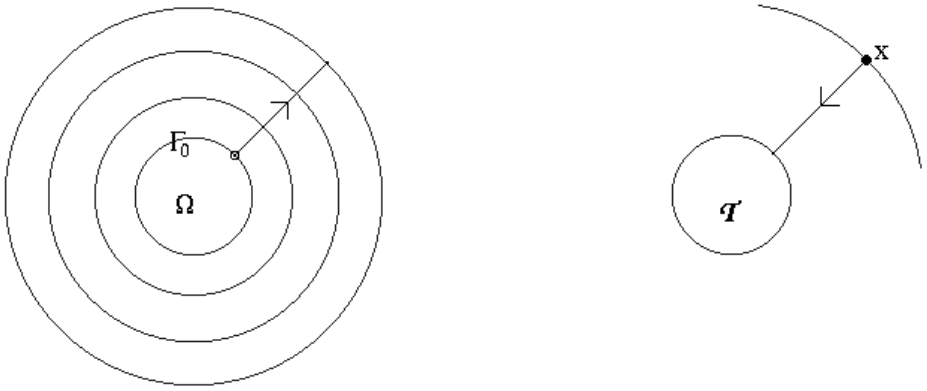


Figura 1.8: equivalenza delle due interpretazioni

Con l'aggiunta delle ovvie condizioni al bordo, otteniamo in conclusione

$$\begin{cases} c(x)|\nabla T(x)| = 1 & x \in \mathbb{R}^n \backslash \mathcal{T} \\ T(x) = 0 & x \in \partial\mathcal{T} \end{cases} \tag{1.19}$$

che riconosciamo essere esattamente l'equazione (1.7) che descrive l'evoluzione dei fronti, identificando il *target* con il fronte iniziale. Da ora in poi quindi indicheremo il *target* anche con Ω, in analogia con quanto studiato nel paragrafo precedente.

In sostanza abbiamo mostrato che la curva di livello $\{T(x) = t\}$, oltre a rappresentare il profilo del fronte al tempo t, rappresenta anche l'insieme dei punti x che raggiungono il *target* esattamente nel tempo t.
Inoltre, come è mostrato in [14], la traiettoria percorsa da un singolo punto del fronte iniziale $x(\bar{s}, t)$ è soluzione di

$$\begin{cases} \dot{y}(t) = -c(y(t))\alpha^*(t) \\ y(0) = x \end{cases}$$

dove α^* è il *controllo ottimo*. Di conseguenza essa coincide con la traiettoria ottima che porta il punto $x(\bar{s}, t)$ a raggiungere il *target* nel tempo t. La Figura 1.8 mostra un facile esempio in cui un fronte circolare si espande a velocità costante. In questo caso il fronte evolve uniformemente in tutte le direzioni e le traiettorie ottime (cioè le curve che è più conveniente seguire per raggiungere il *target*) sono rette.

Se Γ_0 e $c(x)$ sono generici, la ricostruzione delle traiettorie ottime è altrettanto semplice. Infatti, dalla (1.18) ricaviamo che le traiettorie ottime sono

ortogonali alle curve di livello della funzione T.
Questa condizione cessa però di essere vera quando si vuole risolvere un problema di controllo ottimo con un campo b generico.

1.4 Soluzioni di viscosità

Nei paragrafi precedenti abbiamo ricavato, seguendo due punti di vista differenti, l'equazione

$$\begin{cases} c(x)|\nabla T(x)| = 1 & x \in \mathbb{R}^n \backslash \Omega \\ T(x) = 0 & x \in \partial\Omega \end{cases} \tag{1.20}$$

senza mai preoccuparci dei problemi di esistenza e unicità della soluzione ad essa legati. In questo paragrafo affronteremo questo problema.
È noto che questa equazione non ammette in generale l'esistenza di una soluzione *classica*, qui intesa come soluzione di classe C^1 dal momento che nell'equazione compare l'operatore ∇. E questo rimane vero qualsiasi sia la regolarità di c e di Ω. L'esempio 2 del capitolo 5 mostra un valido controesempio in $\mathbb{R}^2$, in cui si ha:

$$\Gamma_0 = (\sin(3\sin(\theta)), \cos(\theta)), \quad \theta \in [0, 2\pi) \qquad \text{e} \qquad c(x, y) \equiv 1$$

ma

$$T(x, y) \notin C^1(\mathbb{R}^2)$$

D'altro canto è altrettanto noto che l'equazione può ammettere infinite soluzioni *deboli*, intese come soluzioni derivabili quasi ovunque che verificano l'equazione in ogni punto di regolarità.
Nella Figura 1.9 sono disegnate a titolo di esempio alcune soluzioni dell'equazione unidimensionale

$$\begin{cases} |T'(x)| = 1 & x \in \mathbb{R} \backslash [-1, 1] \\ T(x) = 0 & x = \pm 1 \end{cases} \tag{1.21}$$

Nel 1983, M.G.Crandall e P.L.Lions [8] hanno risolto il problema introducendo la nozione di soluzione di *viscosità*. Essi hanno individuato le proprietà che caratterizzano una e una sola soluzione dell'equazione (1.20) in modo tale da poter formulare un teorema di esistenza e unicità. Presentiamo qui di seguito una definizione di soluzioni di *viscosità* diversa dall'originale, più semplice ma perfettamente equivalente:

Definizione 1.2 *$u(x) \in C^0(\Omega \subseteq \mathbb{R}^n)$ è una soluzione di* viscosità *di*

$$H(x, u, \nabla u) = 0, \quad x \in \Omega, \qquad H : \Omega \times \mathbb{R} \times \mathbb{R}^n \to \mathbb{R} \tag{1.22}$$

se per ogni $\phi \in C^1(\Omega)$ e

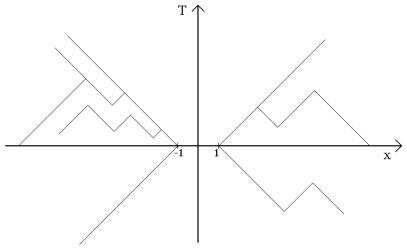


Figura 1.9: soluzioni dell'equazione (1.21)

1. *per ogni x_0 punto di massimo locale per $u - \phi$ si ha*

$$H(x_0, u(x_0), \nabla\phi(x_0)) \leq 0 \qquad (u \text{ è una sottosoluzione})$$

2. *per ogni x_0 punto di minimo locale per $u - \phi$ si ha*

$$H(x_0, u(x_0), \nabla\phi(x_0)) \geq 0 \qquad (u \text{ è una soprasoluzione})$$

La soluzione di *viscosità* può essere anche definita come $\lim_{\varepsilon\to 0} u_\varepsilon(x)$ dove $u_\varepsilon(x)$ è l'*unica* soluzione (classica) del problema perturbato

$$-\varepsilon\triangle u + H(x, u, \nabla u) = 0$$

Dal momento che fisicamente il termine $\triangle u$ rappresenta generalmente la viscosità di un fluido, questo modo di determinare la soluzione di (1.22) viene chiamato metodo della "viscosità evanescente", da cui deriva il nome "soluzione di *viscosità*" (cfr. [1] e [3]). Enunciamo ora il seguente

Teorema 1.3 (***Esistenza e unicità della soluzione di*** **viscosità**) *Sia Ω un aperto limitato di $\mathbb{R}^n$ con frontiera regolare e $H : \Omega \to \mathbb{R}$ tale che*

$$H \in C^0(\Omega), \quad H \text{ convessa}, \quad H(0) = 0 \quad e \quad H \geq 0$$

Sia inoltre $c : \Omega \to \mathbb{R}$ tale che

$$c \in C^0(\overline{\Omega}), \quad c > 0$$

Allora esiste un'unica soluzione di viscosità *u del problema*

$$\begin{cases} c(x)H(\nabla u) = 0 & x \in \Omega \\ u(x) = 0 & x \in \partial\Omega \end{cases} \tag{1.23}$$

La dimostrazione del teorema 1.3 si può trovare in [8].
Nel caso in cui Ω sia un aperto illimitato, è possibile dimostrare un analogo risultato definendo una successione $\Omega_n = \Omega \cap B(0,n)$ e passando al limite per $n \to +\infty$.

Corollario 1.4 *Se $c(x)$ verifica le ipotesi precedenti ed inoltre $c(x)$ è lipschitziana, allora l'equazione* (1.20) *ammette un'unica soluzione di* viscosità *T, ed essa è localmente lipschitziana in $\mathbb{R}^n \backslash \Omega$.*

A questo punto abbiamo tutti gli strumenti necessari per dare una formulazione precisa del teorema "ponte" utilizzato nel paragrafo 1.1, che permette il collegamento tra il caso evolutivo e quello stazionario.

Teorema 1.5 *Sia*

$$c \in Lip(\mathbb{R}^n) \cap L^\infty(\mathbb{R}^n) \quad e \quad c(x) > 0 \ \ \forall x \in \mathbb{R}^n \backslash \Omega$$

Supponiamo inoltre che Ω abbia una frontiera regolare.
Allora l'equazione (1.7) *ammette un'unica soluzione di* viscosità *T ed essa è localmente lipschitziana in $\mathbb{R}^n \backslash \Omega$. Inoltre, la funzione*

$$u(x,t) = T(x) - t$$

è l'unica soluzione di viscosità *dell'equazione*

$$\begin{cases} u_t + c(x)|\nabla u| = 0 & x \in \mathbb{R}^n \backslash \Omega, \ t > 0 \\ u(x,0) = T(x) & x \in \mathbb{R}^n \backslash \Omega \end{cases}$$

La dimostrazione si può ottenere facilmente dai risultati ottenuti in [14].

Elenchiamo alcune importanti proprietà della soluzione di *viscosità*:
Sia $u \in C^0(\Omega)$ la soluzione di *viscosità* di (1.23). Allora

1. Se $u \in C^1(\Omega)$ allora u coincide con l'unica soluzione *classica* della (1.23).

2. Se la (1.23) ammette una soluzione *classica*, allora essa coincide con u.

3. Se $u \in Lip(\Omega)$ allora u verifica la (1.23) per quasi ogni $x \in \Omega$.

4. Sia $\widetilde{u} \in C^0(\Omega)$ una sottosoluzione di (1.23) tale che per ogni v sottosoluzione si abbia
$$\widetilde{u} \geq v.$$
Allora $\widetilde{u}$ è una soprasoluzione di (1.23) e quindi $\widetilde{u} = u$.

Inoltre valgono le seguenti proposizioni:

Proposizione 1.6 (***Proprietà di stabilità***) *Sia* $\{u_n\}_{n\in\mathbb{N}} \in C^0(\Omega)$ *una successione di funzioni tale che* u_n *è soluzione di* viscosità *del problema*

$$H_n(x, u_n, \nabla u_n) = 0\,, \quad x \in \Omega$$

per ogni $n \in \mathbb{N}$*. Supponiamo inoltre che* u_n *e* H_n *convergano localmente uniformemente rispettivamente a* u *e* H*. Allora* u *è soluzioni di* viscosità *di*

$$H(x, u, \nabla u) = 0\,, \quad x \in \Omega$$

Proposizione 1.7 (***Proprietà di ottimalità***) *Sia*

$$x(\bar{s}, t)\,, \quad t \in [0, +\infty)\,, \quad \bar{s} \in [0, S]$$

la curva che il punto $x(\bar{s}, 0) \in \Gamma_0$ *traccia sotto l'azione del campo* $\mathbf{c}(x) = c(x)n$*. Sia inoltre* $T(x)$ *la soluzione di* viscosità *di* (1.20)*.*
Allora

$$t_1 \geq t_2 \Rightarrow T(x(\bar{s}, t_1)) \geq T(x(\bar{s}, t_2)).$$

Quest'ultima proprietà è di fondamentale importanza per la costruzione di tutti gli schemi numerici studiati nei prossimi capitoli. Essa afferma che la funzione T è sempre crescente lungo le curve caratteristiche.
Nel semplice caso dell'esempio (1.21), quest'ultima proprietà afferma che il tempo impiegato dal fronte a raggiungere un punto $\bar{x} \in \mathbb{R}$ aumenta con l'allontanarsi del punto $\bar{x}$ da $\Gamma_0 = \pm 1$ o - analogamente - che maggiore è la distanza del punto iniziale dal *target* (cioè l'intervallo $[-1, 1]$), maggiore sarà il tempo impiegato per raggiungerlo. Questa proposizione permette quindi di selezionare facilmente la soluzione di *viscosità* tra tutte quelle disegnate in Figura 1.9. Essa infatti è

$$T(x) = |x| - 1\,, \quad x \in \mathbb{R}\backslash[-1, 1]$$

cioè è la soluzione che "cresce sempre" allontanandosi dal fronte iniziale e, in accordo con la proprietà 4, è quella che "sta al di sopra" di tutte le altre. Inoltre, si vede facilmente che per la soluzione di *viscosità* di un problema "esterno", si ha sempre

$$\lim_{|x|\to+\infty} T(x) = +\infty.$$

Queste proprietà della soluzione di *viscosità* mostrano che essa coincide con la soluzione fisicamente rilevante. Questa soluzione infatti, deve verificare

alcune condizioni, come ad esempio quella di non introdurre nel sistema fisico informazioni che non sono già contenute nei dati iniziali, ma dare però la possibilità alla soluzione di perdere alcune informazioni rendendo quindi irreversibile l'evoluzione. Ad esempio, se prendiamo come nuovo dato iniziale la soluzione al tempo $\bar{t} \in [0, +\infty)$ e facciamo evolvere il sistema al contrario (verso il passato) fino al tempo $t = 0$ potremmo non ottenere il dato iniziale da cui eravamo partiti (esattamente ciò che accade nella realtà). L'esempio presentato in Figura 1.5 mostra un caso di questo genere. Dopo la fusione dei due fronti il sistema non è più in grado di risalire agli esatti dati iniziali.

Inoltre la nozione di soluzione di *viscosità* risolve il problema dell'autointersezione del fronte accennato nel paragrafo 1.1. Definiamo a questo proposito l'insieme

$$\Omega_t = \Big\{ x \in \mathbb{R}^n \ : \ v(x) \leq 1 - e^{-t} \Big\} = \Big\{ x \in \mathbb{R}^n \ : \ T(x) \leq t \Big\}.$$

Abbiamo già osservato che

$$\Gamma_t = \partial\Omega_t \,, \quad \forall t \geq 0.$$

La continuità della funzione T (assicurata dal corollario 1.4) e la proprietà di *ottimalità*, assicurano che

$$s \leq t \ \Rightarrow \ \Omega_s \subseteq \Omega_t \,, \quad \forall s, t \geq 0.$$

Questo vuol dire che le curve di livello della funzione T sono curve chiuse "concentriche", che non possono quindi intersecarsi.

1.5 Metodi di approssimazione "classici"

In questo paragrafo richiameremo brevemente gli schemi numerici più comunemente usati per approssimare la soluzione di *viscosità* dell'equazione (1.20). Accenniamo già da ora che questi schemi numerici sono sufficientemente robusti per trattare bene anche casi in cui c e Ω non verifichino le ipotesi del teorema 1.3.
Si veda inoltre [28] per dei recenti risultati di esistenza e unicità della soluzione di *viscosità* sotto ipotesi molto più deboli per la funzione c.

1.5.1 Metodi di approssimazione semilagrangiani

Nel paragrafo 1.2 abbiamo visto un'applicazione del PPD che permette di ricavare l'equazione (1.14) per la funzione v. Come mostrato in [2], è possibile

applicare al sistema (1.8) relativo al problema di *tempo minimo* anche la versione *discreta* del PPD, ottenendo l'equazione

$$\begin{cases} v_h(x) = \min\limits_{a\in B(0,1)} \{\beta v_h(x + hb(x,a))\} + 1 - \beta & x \in \mathbb{R}^n \backslash \mathcal{T} \\ v_h(x) = 0 & x \in \partial\mathcal{T} \end{cases} \tag{1.24}$$

dove $\beta = e^{-h}$. Nel caso particolare definito da (1.15) e (1.16), la (1.24) diventa:

$$\begin{cases} v_h(x) = \min\limits_{a\in B(0,1)} \{\beta v_h(x - hc(x)a)\} + 1 - \beta & x \in \mathbb{R}^n \backslash \mathcal{T} \\ v_h(x) = 0 & x \in \partial\mathcal{T} \end{cases} \tag{1.25}$$

Indichiamo con $x_i\,,\ i = 1\ldots, M$, il nodo generico della griglia scelta per l'approssimazione, e con X l'insieme dei nodi della griglia. Possiamo quindi scrivere, per ogni $i = 1, \ldots, M$,

$$\begin{cases} v_h(x_i) = \min\limits_{a\in B(0,1)} \{\beta v_h(x_i - hc(x_i)a)\} + 1 - \beta & x_i \in (\mathbb{R}^n \cap X) \backslash \mathcal{T} \\ v_h(x_i) = 0 & x_i \in \mathcal{T} \cap X \end{cases} \tag{1.26}$$

ponendo per semplicità $v_h = 0$ anche in tutti i nodi *interni* a $\mathcal{T}$.
Per rendere questo schema completamente discreto è necessario approssimare il valore $v_h(x_i - hc(x_i)a)$ tramite un metodo di interpolazione, dal momento che in generale

$$x_i - hc(x_i)a \notin X.$$

Osserviamo che l'interpolazione può essere di ordine arbitrariamente alto. Indichiamo con V la matrice (n-dimensionale) nella quale sono memorizzati i valori

$$v_h(x_i)\,,\ x_i \in X$$

e poniamo $V_i = v_h(x_i)$. Indichiamo inoltre con $G(V)$ l'operatore che provvede all'interpolazione. Ad esempio, nel caso di una interpolazione lineare in $\mathbb{R}^1$, se

$$x_k \leq x_i - hc(x_i)a \leq x_{k+1},$$

si avrà

$$G(V) = \mu V_k + (1 - \mu)V_{k+1}$$

dove μ è un coefficiente da determinare nell'intervallo $[0, 1]$.
Definiamo infine l'operatore F che agisce sulla matrice V nel seguente modo:

$$F(V) = \min_{a\in B(0,1)} \{\beta G(V)\} + 1 - \beta \tag{1.27}$$

In tal modo la (1.26) diventa

$$\begin{cases} V_i = F(V) & \text{se } x_i \in (\mathbb{R}^n \cap X)\backslash\mathcal{T} \\ V_i = 0 & \text{se } x_i \in \mathcal{T} \cap X \end{cases} \tag{1.28}$$

È importante notare che lo schema numerico definito dalla (1.28) *non* è diretto. La tecnica di risoluzione si basa infatti su una procedura di *punto fisso* nella quale si cerca la matrice V^* tale che

$$V^* = F(V^*)$$

La soluzione viene calcolata quindi iterando l'espressione

$$V^{(n)} = F(V^{(n-1)}) \qquad n = 1, 2, \ldots \tag{1.29}$$

scegliendo

$$V^{(0)} = \begin{cases} 0 & \text{se } x_i \in (\mathbb{R}^n \cap X)\backslash\mathcal{T} \\ 1 & \text{se } x_i \in \mathcal{T} \cap X \end{cases} \tag{1.30}$$

I seguenti risultati riassumono le proprietà dello schema appena descritto.

Teorema 1.8 *Sia*

$$c \in Lip(\mathbb{R}^n) \cap L^\infty(\mathbb{R}^n).$$

e supponiamo che $\mathcal{T}$ abbia una frontiera regolare a tratti.
Allora lo schema (1.26) *ammette un'unica soluzione*

$$v_h : (\mathbb{R}^n \cap X)\backslash\mathcal{T} \to [0, 1]$$

lineare a tratti. La procedura di punto fisso, con il dato iniziale (1.30)*, produce una successione monotòna decrescente che converge a V^*.*

Corollario 1.9 *Sia v la soluzione di* viscosità *di* (1.14) *e v_h la soluzione di* viscosità *di* (1.26)*. Sia inoltre*

$$c_{min} = \min_{\mathbb{R}^n\backslash\mathcal{T}} c(x)$$

e sia $k \leq c_{min}h$ una costante positiva.
Allora esistono due costanti positive C_1 e C_2 tali che

$$\|v - v_h\|_\infty \leq C_1 h + C_2 k$$

e

$$d_H(\Omega_t, \hat{\Omega}_t) \leq C_1 h + C_2 k$$

dove d_H denota la distanza di Hausdorff tra due insiemi e $\Omega_t, \hat{\Omega}_t$ sono rispettivamente le regioni delimitate dalle curve di livello relative al valore $1 - e^{-t}$ delle funzioni v e v_h.

Per la dimostrazione di questi risultati rimandiamo a [13], [2] e [15].
È importante notare che il coefficiente di contrazione di F è esattamente β e che

$$\lim_{h\to 0} \beta = 1.$$

Ciò implica che le iterazioni necessarie alla convergenza dello schema numerico aumentano notevolmente al diminuire del passo h.

Confronto con il problema evolutivo

Vogliamo ora sottolineare l'analogia tra lo schema numerico appena ricavato e l'equivalente schema semilagrangiano che è possibile ottenere a partire dal problema evolutivo, cioè dall'equazione (1.5). Introduciamo i passi di discretizzazione spaziale e temporale Δx e Δt, e introduciamo la seguente notazione:

$$u_i^n = u(x_i, t_n) = u(i\Delta x, n\Delta t)\,, \quad i, n \in \mathbb{Z}$$

Sfruttando i numerosi risultati teorici noti per questa equazione, e in particolare l'espressione della soluzione data dalla formula di Hopf-Lax (vedi ad esempio [11]) è possibile ricavare il seguente schema numerico esplicito a un passo:

$$u_i^{n+1} = \min_{a\in B(0,1)} \Big\{ u^n(x_i - ac(x_i)\Delta t) \Big\}.$$

Interpolando la funzione u^n come nel caso stazionario, otteniamo

$$u_i^{n+1} = \min_{a\in B(0,1)} \Big\{ G(u^n) \Big\} \tag{1.31}$$

A questo punto risulta chiara l'analogia con l'equazione (1.29). Le iterazioni di punto fisso si comportano approssimativamente come dei passi di avanzamento in tempo. Per questo motivo il parametro h è spesso considerato un passo discretizzazione temporale fittizio.

Osservazione 1.10 *Questa analogia ridimensiona il vantaggio della versione stazionaria (equazione* (1.7)*) rispetto a quella evolutiva (equazione* (1.5)*). Infatti il costo computazionale della* (1.31) *è paragonabile a quello della* (1.29)*. Non bisogna però dimenticare che esistono per la versione stazionaria delle tecniche di accelerazione della convergenza che non possono essere applicate al caso evolutivo. Inoltre, nel caso stazionario esiste il vantaggio concreto di un risparmio di memoria utilizzata dal calcolatore.*

1.5.2 Metodi di approssimazione alle differenze finite

Un approccio di tipo "punto fisso" è possibile anche con un approssimazione dell'equazione basata sulle differenze finite.
Si inizializza il valore dei nodi ponendo

$$T(x_i) = 0 \qquad \forall\, x_i \in \mathcal{T} \cap X$$

e $T = +\infty$ (da intendersi come il massimo numero accettato dal calcolatore) altrove. Si procede poi con successive iterazioni, calcolando il valore di ogni nodo a partire dal valore dei nodi vicini, in accordo con un'approssimazione del gradiente ottenuta con le differenze finite. Una possibile discretizzazione che garantisce una corretta correzione *up-wind* sarà studiata e analizzata nel prossimo capitolo. Si veda anche [10] per un'analisi degli schemi numerici alle differenze finite per l'analogo problema evolutivo.

È da notare che i due metodi appena descritti sono semplici da implementare ma hanno in generale un alto costo computazionale ed in dimensione $n \geq 3$ necessitano di una implementazione su macchine parallele. Nel 1996 J. A. Sethian , sulla base di un precedente lavoro di J. N. Tsitsiklis, ha introdotto i cosiddetti metodi con *narrow band*, con l'intento di creare nuovi algoritmi per l'evoluzione dei fronti molto più veloci di quelli esistenti. Essi possono essere applicati sia all'equazione evolutiva (1.5) (vedi ad esempio [24]) che all'equazione stazionaria, come vedremo dettagliatamente nel prossimo capitolo.

Capitolo 2

Fast marching method

In questo capitolo illustreremo il *Fast Marching Method* (FMM) introdotto da J. A. Sethian nel 1996 (vedi ad esempio [25] e [24]). Il FMM è un metodo locale basato su una approssimazione alle differenze finite che produce un algoritmo molto veloce per la risoluzione numerica di equazioni del tipo

$$\begin{cases} c(x)|\nabla T(x)| = 1 & x \in \mathbb{R}^n \backslash \Omega \\ T(x) = 0 & x \in \partial\Omega \end{cases} \tag{2.1}$$

sotto l'ipotesi che $c(x) > 0$ o $c(x) < 0$ per ogni $x \in \mathbb{R}^n \backslash \Omega$, qualsiasi sia l'insieme $\Omega \subset \mathbb{R}^n$ (anche non connesso). Il FMM può essere anche applicato per risolvere la versione "interna" dello stesso problema, nella quale il dominio di definizione della funzione T è Ω.
Da ora in poi limiteremo la nostra attenzione all'equazione (2.1) e supporremo sempre $\mathbb{R}^n = \mathbb{R}^2$ e $c(x) > 0$, cioè considereremo il caso di un fronte bidimensionale che si espande. Tutti i risultati ottenuti possono comunque essere estesi senza difficoltà al caso n-dimensionale.
L'algoritmo permette ovviamente di ricostruire la funzione $T(x)$ solamente in un dominio limitato di $\mathbb{R}^2$, ad esempio $Q\backslash\Omega$ dove Q è un rettangolo abbastanza grande da contenere Ω. Le condizioni da imporre sulla frontiera di Q saranno discusse in seguito.

2.1 Idee di base

Per descrivere il metodo in maniera semplice è bene fissare le idee con un esempio. Supponiamo di avere come fronte iniziale un cerchio di raggio 1 centrato nell'origine e un campo di velocità $c(x)$ costantemente uguale a 1. Ci aspettiamo quindi di dover ricostruire la funzione $T(x, y) = |(x, y)| - 1$ (vedi Figura 2.1). Le sue curve di livello rappresenteranno l'evoluzione del

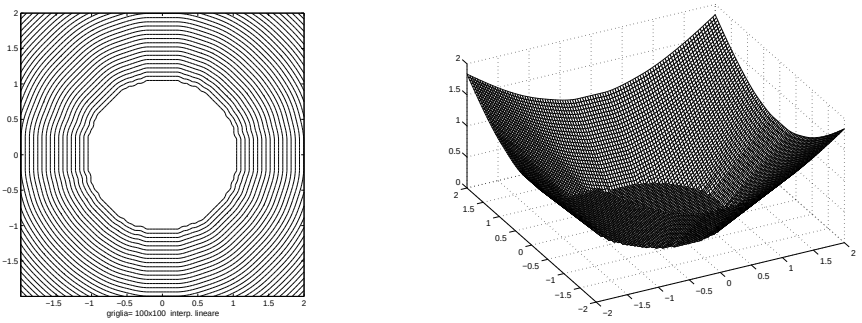


Figura 2.1: funzione $T(x, y) = |(x, y)| - 1$ e sue curve di livello

fronte ad ogni tempo. L'idea di base del FMM consiste nel suddividere i nodi della griglia in tre gruppi: nodi *far*, nodi *accepted* e nodi *narrow band*. I nodi *far* sono i nodi che ancora non sono mai stati calcolati. I nodi *accepted* sono i nodi che sono già stati calcolati e il cui valore è ormai definitivo. La *narrow band* è invece costituita da tutti quei nodi che sono stati calcolati almeno una volta ma ancora non sono stati accettati definitivamente (perché il loro valore potrebbe ancora cambiare).
L'algoritmo è costruito in modo tale da *mimare* l'evoluzione del fronte seguendo la sua evoluzione istante per istante. Infatti i nodi *far* costituiscono la regione di spazio che ancora non è stata raggiunta dal fronte, i nodi *accepted* costituiscono la regione di spazio che è stata già raggiunta dal fronte e la *narrow band* forma, iterazione dopo iterazione, una sorta di *cornice* dei nodi *accepted* costituendo l'insieme dei nodi che stanno per essere raggiunti dal fronte (vedi Figura 2.2). L'algoritmo inizia etichettando come nodi *accepted* solamente i nodi che costituiscono il fronte iniziale $\Gamma_0 = \partial\Omega$ ed i nodi interni ad esso (che non devono essere calcolati). L'algoritmo ha termine quando tutti i nodi di Q sono diventati *accepted*.

2.2 Discretizzazione dell'equazione

Consideriamo un quadrato Q abbastanza grande da contenere Ω, nel quale intendiamo ricostruire la funzione T e costruiamo una griglia strutturata di $N \times N$ nodi. Indicheremo con Δx e Δy i passi di discretizzazione. L'insieme dei nodi sarà quindi

$$\Big\{(i\Delta x, j\Delta y) \in \mathbb{R}^2, \quad i = 1, \ldots, N \quad j = 1, \ldots, M\Big\}$$

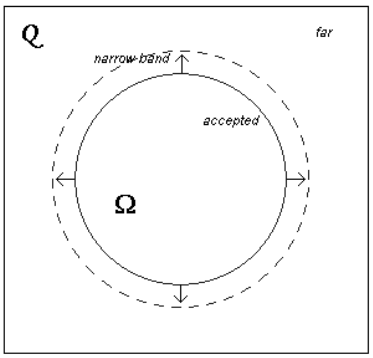


Figura 2.2: nodi *far*, *narrow band* e *accepted*

Indicheremo con (x_i, y_j) il nodo $(i\Delta x, j\Delta y)$ e indicheremo con $T_{i,j}$ e $c_{i,j}$ rispettivamente il valore delle funzioni T e c nel nodo (x_i, y_j). Riscriviamo ora

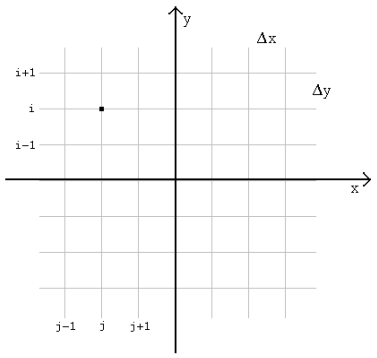


Figura 2.3: griglia strutturata

l'equazione (2.1) esplicitando l'operatore ∇ ed elevando al quadrato ambo i membri:

$$T_x^2 + T_y^2 = \frac{1}{c^2(x,y)} \tag{2.2}$$

Approssimiamo ora le derivate parziali della funzione T con le differenze finite del primo ordine, scegliendo per semplicità $\Delta x = \Delta y$:

$$T_x(x_i, y_j) \approx \frac{T_{i,j} - T_{i-1,j}}{\Delta x} \qquad T_y(x_i, y_j) \approx \frac{T_{i,j} - T_{i,j-1}}{\Delta x}$$

o anche

$$T_x(x_i, y_j) \approx \frac{T_{i+1,j} - T_{i,j}}{\Delta x} \qquad T_y(x_i, y_j) \approx \frac{T_{i,j+1} - T_{i,j}}{\Delta x}$$

Per costruire uno schema numerico che garantisca una corretta correzione *upwind* a partire dall'equazione (2.1) e approssimare correttamente la soluzione di *viscosità* è necessaria una opportuna scelta dell'approssimazione delle derivate. Una buona discretizzazione dell'equazione è la seguente:

$$\left(\max\left\{ \max\left\{ \frac{T_{i,j} - T_{i-1,j}}{\Delta x}, 0 \right\}, -\min\left\{ \frac{T_{i+1,j} - T_{i,j}}{\Delta x}, 0 \right\} \right\} \right)^2 +$$
$$+ \left(\max\left\{ \max\left\{ \frac{T_{i,j} - T_{i,j-1}}{\Delta x}, 0 \right\}, -\min\left\{ \frac{T_{i,j+1} - T_{i,j}}{\Delta x}, 0 \right\} \right\} \right)^2 = \frac{1}{c_{i,j}^2} \quad (2.3)$$

Osserviamo che questa discretizzazione è simile alla seguente:

$$\frac{1}{\Delta x^2} \operatorname{minmod}^2(T_{i+1,j} - T_{i,j},\ T_{i,j} - T_{i-1,j}) +$$
$$+ \frac{1}{\Delta x^2} \operatorname{minmod}^2(T_{i,j+1} - T_{i,j},\ T_{i,j} - T_{i,j-1}) = \frac{1}{c_{i,j}^2} \quad (2.4)$$

dove

$$\operatorname{minmod}(a, b) = \begin{cases} a & \text{se } |a| < |b| \text{ e } ab > 0 \\ b & \text{se } |b| < |a| \text{ e } ab > 0 \\ 0 & \text{se } ab \leq 0 \end{cases}$$

È da notare che la discretizzazione (2.4) pone 0 come valore della derivata nel caso in cui la derivata destra e la derivata sinistra siano di segno discorde, cosa che non accade nella (2.3), dove invece viene scelta la derivata con il valore massimo in modulo. Inoltre, anche nel caso in cui si ha $ab > 0$ i due schemi non coincidono.

Per capire la ragione della validità della (2.3) è utile ragionare *a posteriori*, cioè supponendo di conoscere la soluzione esatta $T(x, y)$ e verificare che questa effettivamente risolva la (2.3). Ragioniamo a tal proposito ancora sull'esempio del paragrafo precedente (fronte iniziale = $B(0, 1)$ e $c \equiv 1$ per ogni x, Figura 2.1) focalizzando l'attenzione su un nodo (x_i, y_j) posto sulla semiretta delle x positive (vedi la Figura 2.4). La situazione è la seguente:

$$T_{i+1,j} = T_{i-1,j} \qquad T_{i,j+1} > T_{i,j} > T_{i,j-1} \qquad T_{i,j} < T_{i+1,j} \qquad T_{i,j} < T_{i-1,j}$$

La (2.3) diventa quindi

$$\left(\frac{T_{i,j} - T_{i,j-1}}{\Delta x} \right)^2 = \frac{1}{c_{i,j}^2} \quad (2.5)$$

da cui ricaviamo

$$(T_{i,j} - T_{i,j-1}) = \pm \frac{\Delta x}{c_{i,j}}$$

E poiché $T_{i,j} - T_{i,j-1} > 0$ e $c_{i,j} > 0$ possiamo selezionare una delle due soluzioni e scrivere:

$$T_{i,j} = T_{i,j-1} + \frac{\Delta x}{c_{i,j}}. \tag{2.6}$$

Nel caso in cui il nodo (x_i, y_j) fosse sulla semiretta delle negative, ricaveremmo analogamente

$$T_{i,j} = T_{i,j+1} + \frac{\Delta x}{c_{i,j}}.$$

Questo esempio mostra chiaramente la presenza di una correzione *up-wind* nello schema numerico. Infatti, esso è costruita in modo tale da calcolare il valore di $T_{i,j}$ sfruttando solamente il valore di T nei nodi che "guardano" verso il fronte. Poiché, come abbiamo già accennato, il calcolo dei nodi avviene seguendo il profilo del fronte mentre evolve, questo tipo di discretizzazione permette all'algoritmo di calcolare il valore di $T_{i,j}$ utilizzando solamente i nodi che sono già stati calcolati.
La discretizzazione *minmod* invece, è costruita in modo tale da selezionare la direzione lungo la quale la funzione subisce una variazione minore, per ridurre brusche oscillazioni nella soluzione numerica.
L'esempio appena descritto mostra la situazione più semplice che si può verificare, cioè quella nella quale un solo addendo, sui due presenti nella (2.3) a primo membro, è diverso da zero. Studiamo ora il caso in cui entrambi gli addendi sono diversi da zero, sempre nel caso particolare del fronte circolare.

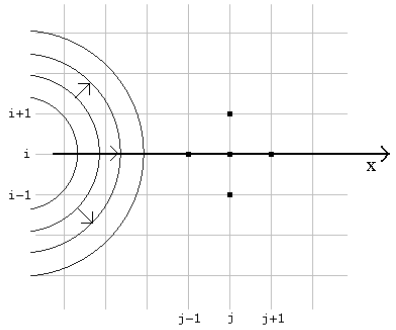


Figura 2.4: fronte circolare che evolve a velocità costante

Consideriamo ora un nodo (x_i, y_j) nel secondo quadrante. Esso sarà raggiunto dal fronte da sud-est (vedi Figura 2.5). Di conseguenza, ragionando

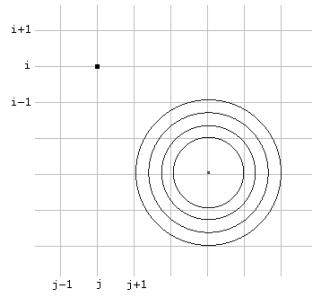


Figura 2.5: fronte circolare che evolve a velocità costante

analogamente al caso precedente, avremo

$$T_{i,j} < T_{i+1,j} \qquad T_{i,j} < T_{i,j-1} \qquad T_{i,j} > T_{i-1,j} \qquad T_{i,j} > T_{i,j+1}$$

È facile verificare che la (2.3) diventa

$$\left(\frac{T_{i,j} - T_{i-1,j}}{\Delta x}\right)^2 + \left(\frac{T_{i,j+1} - T_{i,j}}{\Delta x}\right)^2 = \frac{1}{c_{i,j}^2}$$

da cui

$$T_{i,j} = \frac{T_{i-1,j} + T_{i,j+1} \pm \sqrt{2\left(\frac{\Delta x}{c_{i,j}}\right)^2 - \left(T_{i-1,j} - T_{i,j+1}\right)^2}}{2} \tag{2.7}$$

Anche in questo caso le condizioni

$$T_{i,j} > T_{i-1,j} \qquad T_{i,j} > T_{i,j+1}$$

mi obbligano a selezionare una sola delle due soluzioni della (2.7) e precisamente la soluzione ottenuta scegliendo il segno "+". Infatti

$$\left\{ \begin{array}{l} T_{i,j} > T_{i-1,j} \\ T_{i,j} > T_{i,j+1} \end{array} \right. \quad \Rightarrow \quad T_{i,j} > (T_{i-1,j} + T_{i,j+1})/2.$$

Un'altra possibile discretizzazione dell'equazione (2.1) è la seguente:

$$\left(\max\left\{\frac{T_{i,j} - T_{i-1,j}}{\Delta x}, 0\right\}\right)^2 + \left(\min\left\{\frac{T_{i+1,j} - T_{i,j}}{\Delta x}, 0\right\}\right)^2 +$$
$$+\left(\max\left\{\frac{T_{i,j} - T_{i,j-1}}{\Delta x}, 0\right\}\right)^2 + \left(\min\left\{\frac{T_{i,j+1} - T_{i,j}}{\Delta x}, 0\right\}\right)^2 = \frac{1}{c_{i,j}^2} \tag{2.8}$$

È importante osservare che in questa discretizzazione c'è l'implicita richiesta che il primo e il secondo addendo non siano contemporaneamente diversi da 0, così come il terzo e il quarto addendo. Per $\Delta x \to 0$ questo significa richiedere (*legittimamente*) che $T_y(x_i, y_j)$ e $T_x(x_i, y_j)$ non siano contemporaneamente maggiori e minori di 0. Numericamente però le cose non funzionano così! Dal punto di vista dell'evoluzione dei fronti, questa richiesta corrisponde ad poter escludere *a priori* che un nodo venga raggiunto dal fronte contemporaneamente da sud e da nord (o da est e da ovest).
Questa condizione può essere troppo restrittiva perché non permette di considerare il caso in cui Ω è un insieme sconnesso (cfr. Figura 2.6, dove

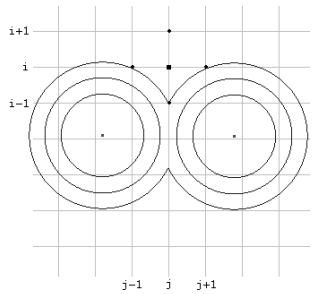


Figura 2.6: nodo raggiunto contemporaneamente da sud-est e sud-ovest

$T_{i,j+1} < T_{i,j}$ e $T_{i,j-1} < T_{i,j}$).

Resta ora da capire come può l'algoritmo riuscire a selezionare la giusta derivata numerica per calcolare il valore di T in un nodo generico, cioè come può individuare la direzione *up-wind* senza (ovviamente) conoscere *a priori* la soluzione T. Come vedremo il punto chiave sta nell'inizializzazione dei valori dei nodi.

2.3 Descrizione dell'algoritmo

Descriviamo ora in maniera sintetica l'algoritmo per poi commentarlo più dettagliatamente. Iniziamo con la seguente

Definizione 2.1 *Sia (x_i, y_j) un generico nodo. L'insieme dei* vicini *di questo nodo è:*

$$V = \Big\{(x_{i+1}, y_j), (x_{i-1}, y_j), (x_i, y_{j+1}), (x_i, y_{j-1})\Big\}$$

Questa definizione si può estendere anche a nodi di una griglia di $\mathbb{R}^n$, $n \geq 3$. Ad esempio, per $n = 3$, l'insieme dei vicini del nodo (x_i, y_j, z_k) è

$$V = \Big\{(x_{i+1}, y_j, z_k), (x_{i-1}, y_j, z_k), (x_i, y_{j+1}, z_k), (x_i, y_{j-1}, z_k), \\ (x_i, y_j, z_{k+1}), (x_i, y_j, z_{k-1})\Big\}. \tag{2.9}$$

Passiamo ora alla descrizione dell'algoritmo.

Inizializzazione (vedi Figura 2.7)

1. Si individuano i nodi che costituiscono il fronte iniziale che vengono etichettati come *accepted*, assegnando loro il valore $T = 0$.

2. Si individuano i nodi della *narrow band* iniziale, cioè i nodi *vicini* al fronte, esterni se $c > 0$ o interni se $c < 0$. Essi vengono etichettati come *narrow band*, assegnando loro il valore $T = \frac{\Delta x}{c}$.

3. Si etichettano come *far* tutti i rimanenti nodi della griglia assegnando loro il valore $T = +\infty$ (da intendersi come il massimo numero accettato dal calcolatore).

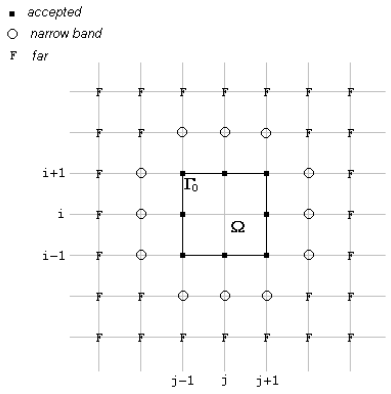


Figura 2.7: inizializzazione nel caso $c > 0$

Nel paragrafo 2.5 è spiegato nel dettaglio come eseguire queste operazioni.

Ciclo principale

1. Si individuano, tra i nodi della *narrow band*, quello cui è assegnato il minimo valore per T. Indicheremo questo nodo con A.

2. Si etichetta il nodo A con *accepted* e lo si rimuove dalla *narrow band*.

3. Si etichettano come *attivi* i *vicini* di A che non sono *accepted*. Se tra questi ci sono nodi *far*, li si trasferisce nella *narrow band*.

4. Si calcola (o ricalcola) il valore di T nei nodi *attivi* risolvendo l'equazione di secondo grado (2.3) scegliendo la soluzione più grande tra le due.

5. Se la *narrow band* non è vuota si ritorna al punto 1.

Commentiamo ora l'algoritmo sviluppandolo su un esempio semplice. Supponiamo di avere un fronte Γ_0 con configurazione iniziale puntiforme che si propaga a velocità $c \equiv 1$ per ogni $x \in Q \backslash \Gamma_0$. L'inizializzazione consiste nel-

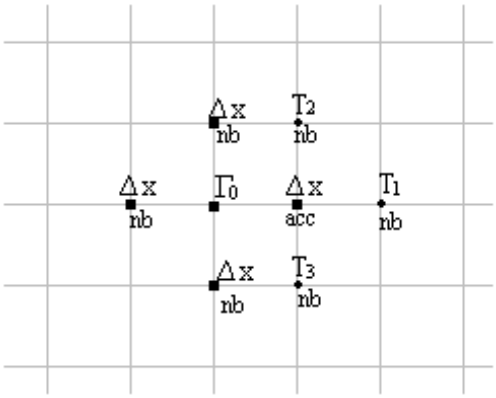


Figura 2.8: FMM con fronte iniziale puntiforme e velocità costante

l'assegnare al nodo corrispondente a Γ_0 il valore $T = 0$ e l'etichetta *accepted*. Assegnare poi ai suoi quattro vicini il valore $T = \Delta x$ e l'etichetta *narrow band*. A tutti gli altri nodi assegnare il valore $T = +\infty$ e l'etichetta *far*. Viene ora convertito ad *accepted* il nodo nella *narrow band* con il minimo valore per T, in questo caso uno qualsiasi dei quattro nodi della *narrow band* (vedi Figura 2.8).
È importante far notare che il valore del nodo convertito ad *accepted* non sarà *mai più* calcolato. Fisicamente questo corrisponde all'aver accettato che il fronte Γ_t abbia raggiunto quel nodo.

A questo punto la *narrow band* si "allarga" a partire da questo nodo inglobando i suoi tre vicini non *accepted*, ai quali sarà assegnato rispettivamente il valore T_1, T_2 e T_3. Grazie alla natura *up-wind* dello schema numerico e all'assegnazione del valore $+\infty$ ai nodi *far*, questo valore sarà calcolato esclusivamente utilizzando i valori dei nodi già calcolati (nel nostro esempio, solamente i valori $T = 0$ e $T = \Delta x$). I nodi *far* saranno automaticamente esclusi dal calcolo.
L'algoritmo riparte convertendo ad *accepted* un altro nodo della *narrow band* con valore Δx e ricalcolandone i vicini non *accepted*. È da notare che i valori T_1, T_2 e T_3 sono sicuramente tutti maggiori di Δx (come dimostreremo nel prossimo capitolo e come ci si aspetta dall'interpretazione di T come tempo di primo arrivo). Di conseguenza i tre nodi con questi valori vengono "messi in attesa" e non diventeranno *accepted* finché non diventeranno *accepted* tutti i nodi con valore Δx.
Questo modo di aggiornare la *narrow band*, cancellando sempre il valore più piccolo, fa sì che non ci siano mai grandi differenze tra i valori dei nodi presenti in essa. La *narrow band* è quindi un'approssimazione delle curve di livello della funzione $T(x, y)$.

È da notare che una volta terminato il calcolo, nei nodi interni a Γ_0 sarà rimasto il valore $T = +\infty$. È consigliabile quindi assegnare a tutti questi nodi il valore $T = 0$ o $T = -1$ in modo tale che essi non diano problemi nel disegnare le curve di livello della funzione $T(x, y)$.

2.3.1 Osservazioni sul costo computazionale

Supponiamo per il momento che l'algoritmo possa essere portato a termine e che converga alla soluzione di *viscosità* dell'equazione (2.1) (questo problema sarà l'oggetto del prossimo paragrafo). Soffermiamoci sul costo computazionale di questo metodo: come si vede facilmente, ogni nodo viene calcolato quando uno dei suoi *vicini* diventa *accepted*. Dal momento che un nodo non può diventare *accepted* più di una volta e che ogni nodo ha quattro vicini, appare chiaro che ogni nodo non potrà mai essere calcolato più di quattro volte. Questo porta a un costo computazionale di $O(N^2)$ dove N^2 è il numero di nodi totale della griglia (nel caso generale p-dimensionale si avrà $O(N^p)$) .
D'altra parte la ricerca del minimo valore di T tra i nodi della *narrow band* richiede al minimo un costo di $O(\ln(N^2))$ (ci si aspetta infatti che all'aumentare del numero complessivo dei nodi cresca anche il numero dei nodi coinvolti istante per istante nella *narrow band*). Di conseguenza l'algoritmo

ha un costo complessivo di $O(N^2 \ln(N^2))$ operazioni. Per una analisi più approfondita rimandiamo a [29] e [24].

2.4 Dimostrazione della validità del metodo

In questo paragrafo dimostreremo che il valore minimo dei nodi della *narrow band* è sicuramente esatto, ed è quindi lecito considerarlo definitivo (cioè farlo diventare *accepted*). Osserviamo che è da considerarsi esatto (o non migliorabile) quel valore che sicuramente non potrà essere sostituito, ad un passo successivo del procedimento, con un valore strettamente minore. Il valore calcolato sarà comunque affetto da un errore dovuto all'approssimazione numerica.

Supponiamo, senza perdere di generalità, che A sia il nodo della *narrow band* con il valore minimo di T (vedi Figura 2.9). Indichiamo con $T(A)$ il

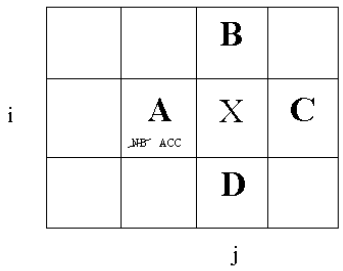


Figura 2.9: matrice dei valori

suo valore. Supponiamo inoltre che B, C e D siano nodi qualsiasi (dunque *accepted*, *narrow band* o *far*), tali che

$$T(B) \leq T(D)$$

(se non fosse così possiamo sempre pensare di scambiare il ruolo di B e D). Sappiamo che a questo punto l'algoritmo ci impone di convertire il nodo A ad *accepted* e di calcolare i suoi vicini non *accepted*. Supponiamo che il nodo X sia tra questi.
Vale la seguente importante

Proposizione 2.2 *Sia T(X) il valore che viene assegnato al nodo* $X = (x_i, y_j)$ *sotto le ipotesi appena descritte. Supponiamo che*

$$c \in Lip(\mathbb{R}^n) \cap L^\infty(\mathbb{R}^n).$$

Indichiamo con L_c la costante di Lipschitz di c. Supponiamo inoltre che

$$c(x) > 0\,, \quad \textit{per ogni} \;\; x \in Q\backslash\Omega.$$

e sia

$$c_{min} = \min_{Q\backslash\Omega} c(x).$$

Supponiamo infine che valga la seguente condizione CFL

$$\Delta x \leq (\sqrt{2}-1)\frac{c_{min}}{L_c}. \tag{2.10}$$

Allora si avrà

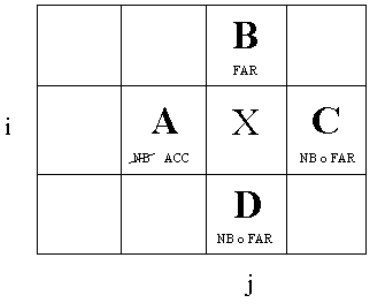
$$T(A) \leq T(X) \leq T(A) + f_X$$

dove

$$f_X \stackrel{def}{=} \frac{\Delta x}{c(X)}$$

Dimostrazione. Utilizzeremo la discretizzazione (2.3) per il calcolo di $T(X)$. È opportuno ricordare che, come abbiamo visto nel paragrafo 2.2, questa discretizzazione fa sì che il valore $T(X)$ venga calcolato tramite il valore di un suo vicino oppure tramite i valori di due suoi vicini.
Dividiamo la dimostrazione in quattro casi:

CASO 1
B è *far*.
C e D sono *narrow band* o *far*.



Per ipotesi sappiamo che $T(B) = +\infty$ e che $T(B) \leq T(D)$. Quindi anche D deve essere *far*. Inoltre si ha

$$T(C) \geq T(A)$$

dal momento che tra i nodi della *narrow band* è stato scelto A per essere convertito ad *accepted*. Anche il nodo X è *far*, perché ancora non è mai stato

calcolato.
Dalla (2.3) ricaviamo che il valore $T(X)$ sarà calcolato tramite la seguente espressione:

$$T(X) = T(A) + f_X \tag{2.11}$$

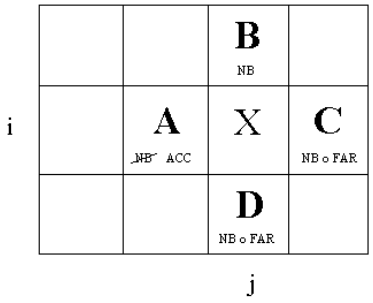
Poiché $f_X > 0$, abbiamo dimostrato che

$$T(X) > T(A). \tag{2.12}$$

Infine, la (2.11) e la (2.12) implicano che :

$$T(A) \leq T(X) \leq T(A) + f_X \tag{2.13}$$

CASO 2
B è *narrow band*.
C e D sono *narrow band* o *far*.



Anche in questo caso il nodo X è *far*. Si ha inoltre

$$T(A) \leq T(B) \qquad T(A) \leq T(C)$$

perché tra i nodi della *narrow band* è stato scelto A per essere convertito ad *accepted*. E ancora, per ipotesi

$$T(B) \leq T(D).$$

Il valore $T(X)$ sarà quindi calcolato utilizzando i valori dei nodi A e B. Dalla (2.3) otteniamo:

$$T(X) = \frac{T(A) + T(B) + \sqrt{2f_X^2 - (T(A) - T(B))^2}}{2} \tag{2.14}$$

Da questa ricaviamo che

$$T(X) \geq \frac{T(A) + T(B)}{2} \geq \frac{T(A) + T(A)}{2} = T(A). \tag{2.15}$$

Dal momento che $T(X)$ è soluzione di

$$(T(X) - T(A))^2 + (T(X) - T(B))^2 = f_X^2$$

segue che

$$(T(X) - T(A))^2 \leq f_X^2$$

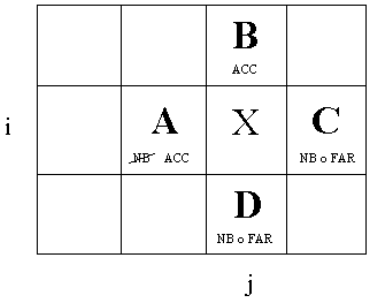
Poiché entrambi i membri dell'ultima disuguaglianza sono positivi, segue che

$$T(X) - T(A) \leq f_X \tag{2.16}$$

In conclusione, unendo i risultati ottenuti, si ottiene

$$T(A) \leq T(X) \leq T(A) + f_X. \tag{2.17}$$

CASO 3
B è *accepted*.
C e D sono *narrow band* o *far*.



In questo caso, il nodo X è già stato calcolato una volta (quando B è diventato *accepted*). Indicheremo il suo valore con $T_{old}(X)$. Il nodo X è quindi nella *narrow band* e deve ora essere ricalcolato perchè vicino del nodo A che è appena diventato *accepted*. Notiamo che il precedente calcolo di $T_{old}(X)$ ha seguito le "regole" del *CASO* 1 o del *CASO* 2. Si ha quindi, dalla (2.13) o dalla (2.17)

$$T(B) \leq T_{old}(X) \leq T(B) + f_X$$

Inoltre

$$T(A) \leq T_{old}(X)$$

perché in caso contrario sarebbe stato scelto X al posto di A come nodo da convertire ad *accepted*. E ancora, sarà

$$T(B) \leq T(A)$$

perché B è diventato *accepted* prima di A.
Riunendo i risultati ottenuti, abbiamo

$$T(B) \leq T(A) \leq T_{old}(X) \leq T(B) + f_X$$

da cui ricaviamo

$$0 \leq T(A) - T(B) \leq f_X. \tag{2.18}$$

Il calcolo del nuovo valore di X, che indicheremo con $T_{new}(X)$, sarà ottenuto sicuramente utilizzando $T(A)$ e $T(B)$ in quanto

$$T(A) \leq T_{old}(X) \qquad T(B) \leq T_{old}(X) \qquad T(A) \leq T(C) \qquad T(B) \leq T(D).$$

Avremo quindi

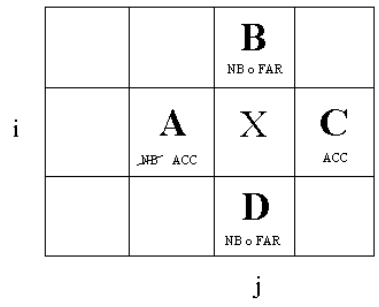
$$\begin{aligned} T_{new}(X) &= \frac{T(A) + T(B) + \sqrt{2f_X^2 - (T(A) - T(B))^2}}{2} \geq \\ &\geq \frac{T(A) + T(B) + f_X}{2} \geq \frac{T(A) + T(A)}{2} = T(A) \end{aligned} \tag{2.19}$$

Inoltre si ha

$$T_{new}(X) \leq \frac{T(A) + T(B) + \sqrt{2} f_X}{2} \leq T(A) + \frac{\sqrt{2}}{2} f_X \leq T(A) + f_X$$

Riunendo i due ultimi risultati si ha la tesi.

CASO 4
B è *narrow band* o *far*.
C è *accepted*.
D è *narrow band* o *far*.



In questo caso il nodo X è già stato calcolato perché vicino di C. Esso è quindi nella *narrow band* e ha il valore $T_{old}(X)$. Inoltre

$$T_{old}(X) \geq T(A)$$

perché in caso contrario sarebbe stato scelto X al posto di A come nodo da convertire ad *accepted*. Inoltre si ha

$$T(C) \leq T(A) \qquad T(B) \leq T(D).$$

Quindi i nodi coinvolti nel calcolo di X saranno C e B oppure solamente il nodo C. Il fatto che A sia diventato *accepted* non ha alcuna influenza e si va quindi a ricadere nel *CASO* 1 o 2. Di conseguenza si avrà

$$T(C) \leq T_{new}(X) \leq T(C) + f_X \leq T(A) + f_X.$$

Inoltre $T_{new}(X)$ viene ricalcolato attraverso gli stessi valori con i quali era stato calcolato $T_{old}(X)$ ed eventualmente altri valori maggiori di $T(A)$. Di conseguenza, dal momento che si era precedentemente ottenuto il valore $T_{old}(X) \geq T(A)$ si avrà ora

$$T_{new}(X) \geq T_{old}(X) \geq T(A).$$

Infine, tutti gli altri casi dove due o più nodi tra B, C e D sono *accepted*, possono essere trattati in modo simile ai casi precedenti. Osserviamo che se D è *accepted*, anche B deve esserlo dal momento che $T(B) \leq T(D)$.

Per completare la dimostrazione, è necessario però dimostrare che l'espressione che compare sotto radice nel calcolo di $T(X)$ tramite il valore di due suoi *vicini*, non può mai essere negativa. In caso contrario infatti il metodo fallirebbe perché non potrebbe fornire un valore per $T(X)$.

CASO 2
Cominciamo con il dimostrare che l'ipotesi (2.10) assicura che

$$\frac{c(Z)}{c(Z')} \leq \sqrt{2} \tag{2.20}$$

per ogni Z e Z' nodi della griglia tali che

$$|Z - Z'| = \Delta x.$$

Infatti, per ipotesi si ha che

$$|c(Z) - c(Z')| \leq L_c |Z - Z'|.$$

Se $|Z - Z'| = \Delta x$, si ha che

$$|c(Z) - c(Z')| \leq L_c \Delta x \leq (\sqrt{2} - 1) c_{min} \leq (\sqrt{2} - 1) c(Z')$$

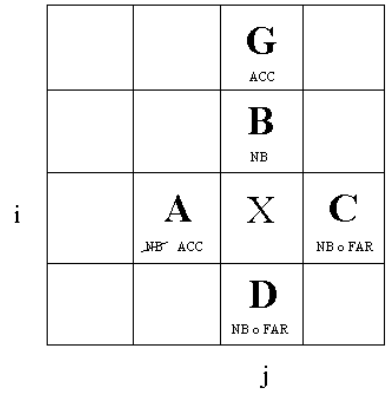
da cui ricaviamo

$$c(Z) - c(Z') \leq (\sqrt{2} - 1)c(Z')$$

e quindi

$$c(Z) \leq \sqrt{2}\, c(Z')$$

come volevasi dimostrare.
Dal momento che il nodo B è nella *narrow band*, un suo *vicino* deve necessariamente essere *accepted*. Sia G questo nodo.



Inoltre si deve avere

$$T(A) \leq T(B)$$

perché è stato scelto A per essere convertito ad *accepted*, e

$$T(G) \leq T(A)$$

perché G è diventato *accepted* prima di A. In base ai risultati ottenuti precedentemente, possiamo affermare che

$$T(G) \leq T(B) \leq T(G) + f_B.$$

da cui ricaviamo

$$T(A) \leq T(B) \leq T(G) + f_B \leq T(A) + f_B$$

e quindi

$$0 \leq T(B) - T(A) \leq f_B. \tag{2.21}$$

Scegliendo $Z = X$ e $Z' = B$ nella (2.20), si ottiene

$$\frac{c(X)}{c(B)} \leq \sqrt{2}$$

e quindi, moltiplicando ambo i membri per Δx,

$$\sqrt{2} f_X \geq f_B.$$

Dalla (2.21) ricaviamo

$$\sqrt{2} f_X \geq T(B) - T(A)$$

e infine

$$2 f_X^2 - (T(B) - T(A))^2 \geq 0$$

CASO 3
In questo caso, la disuguaglianza (2.18) assicura che l'espressione che compare sotto radice sia sempre positiva. ∎

Torniamo ora alla dimostrazione dell'esattezza del valore del nodo che viene convertito ad *accepted*. Chiamiamo T_{min} questo valore. Dal momento che tutti gli altri nodi della *narrow band* hanno un valore più grande di T_{min}, la precedente proposizione assicura che a partire da questi nodi non si potrà mai assegnare ad un nodo un valore più piccolo di T_{min}. Di conseguenza il valore T_{min} può considerarsi esatto, nel senso specificato all'inizio del paragrafo.

2.4.1 Consistenza e convergenza

È facile vedere che, per $\Delta x \to 0$, le discretizzazioni (2.3) e (2.8) ritornano ad essere l'equazione originale, cioè

$$c(x)|T(x)| = 1. \tag{2.22}$$

Di conseguenza entrambe le discretizzazioni possono dirsi consistenti. Inoltre, supponendo per semplicità di scegliere

$$\Delta x = \Delta y$$

è facile vedere che l'ordine di consistenza è $O(\Delta x)$. Infatti, utilizzando la formula di Taylor, si ha

$$T(x + \Delta x, y) = T(x, y) + T_x(x, y)\Delta x + O(\Delta x^2)$$

e analogamente

$$T(x, y + \Delta x) = T(x, y) + T_y(x, y)\Delta x + O(\Delta x^2)$$

Di conseguenza, si ha

$$T_x(x,y) = \frac{T(x+\Delta x, y) - T(x,y)}{\Delta x} + O(\Delta x)$$

e

$$T_y(x,y) = \frac{T(x, y+\Delta x) - T(x,y)}{\Delta x} + O(\Delta x).$$

Il ragionamento è analogo per le derivate sinistre.

Per quanto riguarda la dimostrazione della convergenza, dimostreremo che la soluzione calcolata dal FMM è identica a quella calcolata dallo schema alle differenze finite standard descritto nel paragrafo 1.5.2. La convergenza del FMM seguirà quindi da quella del metodo alle differenze finite standard. Enunciamo il seguente teorema, valido per uno schema numerico generale ed in dimensione qualsiasi.

Teorema 2.3 *Sia* $(V_i)_{i=1,\dots,M}$ *la matrice dei valori dei nodi della griglia n-dimensionale e sia*

$$V_i = F(V_{i-k}, \dots, V_{i+l}) \tag{2.23}$$

lo schema numerico utilizzato per il calcolo di ogni singolo nodo.
Sia $\widehat{V}$ *la matrice ottenuta dall'algoritmo basato sulla tecnica di punto fisso utilizzando lo schema* (2.23) *e sia* $\overline{V}$ *la matrice ottenuta dall'algoritmo basato sulla tecnica della* narrow band*, utilizzando lo* stesso *schema.*
Allora, si ha

$$\overline{V} = \widehat{V}$$

Dimostrazione. Le due matrici coincidono se e solo se

$$\overline{V}_i = F(\overline{V}_{i-k}, \dots, \overline{V}_{i+l}), \quad \forall\ i = 1, \dots, M \tag{2.24}$$

Infatti, se considerassimo la matrice $\overline{V}$ come dato iniziale per lo schema basato sulla tecnica di punto fisso, quest'ultimo sarebbe già arrivato a convergenza. D'altro canto, quando tutti i nodi della griglia sono *accepted*, deve essere necessariamente $\overline{V}_i = F(\overline{V}_{i-k}, \dots, \overline{V}_{i+l})$. Infatti, se così non fosse, vorrebbe dire che il valore di qualche nodo deve ancora essere migliorato, e abbiamo precedentemente dimostrato che questo implica avere ancora qualche nodo nella *narrow band* o tra i nodi *far*. ∎

2.5 Suggerimenti per l'implementazione

2.5.1 Inizializzazione

L'inizializzazione consiste nell'individuare i nodi che costituiscono il fronte iniziale Γ_0 ed i nodi della *narrow band* iniziale. Per fare questo possono

essere sviluppate diverse tecniche, in base al modo con cui viene assegnata l'equazione del fronte iniziale.

1. Il fronte iniziale è la curva di livello di una funzione $f(x, y)$ nota, cioè

$$\Gamma_0 = \Big\{(x, y) \ : \ f(x, y) = 0\Big\}$$

In questo caso si individua l'insieme

$$\widetilde{\Gamma}_0 = \Big\{(x_i, y_j) \ : \ 0 \leq f(x_i, y_j) \leq \varepsilon\Big\}$$

dove ε è un valore positivo, arbitrariamente piccolo, da scegliere in base al passo Δx e alla stessa funzione f. Si inizializzano poi i nodi in $\widetilde{\Gamma}_0$ assegnando loro il valore 0.
Per ogni $X \in \widetilde{\Gamma}_0$, si considera l'insieme V_X dei suoi *vicini*. Gli elementi di V_X vengono poi distinti tra nodi interni a Γ_0 e nodi esterni a Γ_0, semplicemente calcolando il valore di f in quei nodi e osservando se questo valore è positivo (nodi esterni) o negativo (nodi interni). A questo punto si possono inizializzare i nodi interni adiacenti al fronte con il valore -1 (un valore neutro che non entrerà mai in gioco nei calcoli successivi e che non permetterà il propagarsi del fronte al suo interno) e i nodi esterni adiacenti al fronte con il valore $T = \frac{\Delta x}{c}$.
Esistono anche tecniche più sofisticate che permettono di inizializzare i nodi costituenti il fronte senza commettere alcun errore di approssimazione, ma non sono state prese in considerazione in questa tesi.

2. Il fronte iniziale è una curva chiusa del piano di cui è nota l'equazione parametrica, cioè

$$\Gamma_0 = \Big\{(x(s), y(s)), \quad s \in [0, S], \quad (x(0), y(0)) = (x(S), y(S))\Big\}$$

Supporremo, senza perdere di generalità che la curva venga percorsa in senso antiorario.
In questo caso si discretizza l'intervallo $[0, S]$ con un passo Δt (in generale più piccolo del passo in spazio Δx precedentemente scelto) e si percorre il fronte calcolando i punti (x, y) in cui esso cade al tempo $t = 0, \ \Delta t, \ 2\Delta t, \ldots, S$. Si individuano quindi i nodi della griglia più vicini a questi punti e li si inizializza assegnando loro il valore 0. È importante ricordarsi di tenere in memoria le coordinate di tutti i nodi che fanno parte del fronte, nell'ordine con cui sono stati individuati (antiorario) e facendo attenzione che non compaia mai in questa lista due volte lo stesso nodo. Inoltre è fondamentale che i nodi che fanno

parte del fronte siano adiacenti l'uno all'altro in modo da *mimare* la continuità della curva (per questo motivo è bene scegliere il passo in tempo Δt più piccolo del passo in spazio Δx). Così facendo è possibile distinguere in modo relativamente semplice i vicini esterni da quelli interni.
Infatti, si considera un generico nodo (x_i, y_j) del fronte e quindi l'insieme V dei suoi *vicini*. Da questo insieme si eliminano i nodi che fanno già parte del fronte. Si considera quindi un nodo rimasto nell'insieme V, ad esempio il nodo $X = (x_{i+1}, y_j)$ (vedi Figura 2.10). Si individua

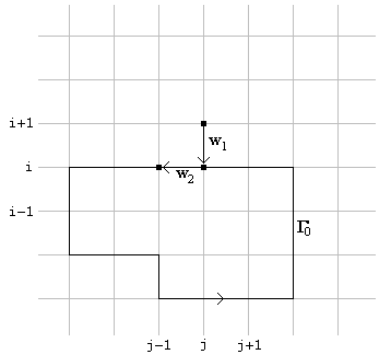


Figura 2.10: distinzione dei nodi interni ed esterni

ora il nodo *successivo* a (x_i, y_j) nella lista (precedentemente memorizzata) dei nodi che compongono il fronte. Sia questo ad esempio il nodo (x_i, y_{j-1}). Definendo ora i tre vettori di $\mathbb{R}^3$

$$\mathbf{w_1} = (x_i, y_j, 0) - (x_{i+1}, y_j, 0)\,, \qquad \mathbf{w_2} = (x_i, y_{j-1}, 0) - (x_i, y_j, 0)$$

e

$$\mathbf{z} = (z_1, z_2, z_3) = \mathbf{w_1} \times \mathbf{w_2}$$

dove $\times$ indica il prodotto vettoriale, si ha che:

$$(x_{i+1}, y_j) \text{ è interno se } z_3 > 0$$

$$(x_{i+1}, y_j) \text{ è esterno se } z_3 < 0$$

se $z_3 = 0$ allora (x_{i+1}, y_j) può essere sia interno che esterno.

In quest'ultimo caso si rende necessaria un'analisi più approfondita. Si può ad esempio partire dal nodo (x_i, y_j) e scorrere nodo per nodo tutta la matrice nelle direzioni N,S,E,W. Se si incontrerà un nodo del fronte

(cioè un nodo con il valore 0) in tutte e quattro le direzioni allora si potrà concludere che il nodo è interno, mentre in caso contrario si potrà concludere che il nodo è esterno. Purtroppo esistono delle curve per cui anche questa analisi più approfondita non riesce a distinguere correttamente i nodi interni da quelli esterni. Nella Figura 2.11 è mostrato un valido controesempio. Fronti iniziali di questo tipo non sono stati

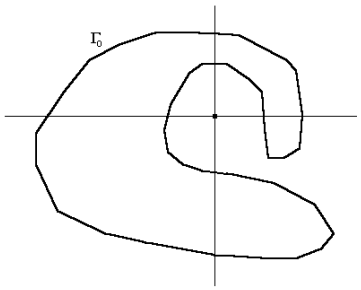


Figura 2.11: controesempio

quindi presi in considerazione.
Si ripetono quindi le operazioni descritte per tutti i *vicini* dei nodi che compongono il fronte, in modo tale da individuare una "cornice" esterna ed una interna di nodi adiacenti ad esso. A questo punto si possono inizializzare i nodi interni adiacenti al fronte con il valore -1 (un valore neutro che non entrerà mai in gioco nei calcoli successivi e che non permetterà il propagarsi del fronte al suo interno) e i nodi esterni adiacenti al fronte con il valore $T = \frac{\Delta x}{c}$.
In conclusione, osserviamo che descrivendo il fronte iniziale tramite equazioni parametriche, otteniamo da una parte una più difficile gestione dell'inizializzazione ma dall'altra otteniamo una maggiore versatilità della scelta del fronte iniziale mantenendo comunque un tempo di calcolo sufficientemente basso.

3. Il fronte iniziale è formato da un singolo punto $O = (\bar{x}, \bar{y})$, cioè

$$\Gamma_0 = \left\{ (\bar{x}, \bar{y}) \right\}$$

Una procedura di inizializzazione molto precisa e compatibile con il FMM (e anche con il FMM-SL descritto nel prossimo capitolo) è la

seguente: si individuano i quattro nodi P_1, P_2, P_3, P_4 che delimitano la cella nella quale cade il punto O. Ad essi viene assegnato il valore

$$T(P_i) = \frac{dist(O, P_i)}{c(P_i)} \qquad i = 1, 2, 3, 4.$$

e l'etichetta *accepted*. Si assegna poi a tutti i *vicini* di questi quattro nodi il valore $T = \frac{\Delta x}{c}$ e l'etichetta *narrow band*.

2.5.2 Gestione delle etichette *far*, *narrow band* e *accepted*

Per tenere in memoria le etichette di tutti i nodi della griglia in modo semplice e veloce, si può procedere nel seguente modo:
All'inizio del procedimento si memorizza una copia della matrice in cui si tengono in memoria i valori di T. Gli elementi di questa seconda matrice saranno 10, 100 o 1000 a seconda che il nodo corrispondente sia rispettivamente *accepted*, *narrow band* o *far*.
È però importante notare che si deve necessariamente memorizzare anche una lista (o meglio un albero) nel quale sono contenuti gli indici e i valori dei nodi che in ogni istante si trovano nella *narrow band*. È qui che si dovrà effettuare la ricerca del minimo valore di T.

2.5.3 Ricerca del minimo valore di T

In questa tesi i valori dei nodi presenti nella *narrow band* sono memorizzati di volta in volta in una lista[1] non ordinata. La ricerca del minimo è quindi ottenuta semplicemente per confronto. Come abbiamo visto nel paragrafo 2.3.1, per raggiungere la complessità di $O(N^2 \ln(N^2))$ è necessario compiere la ricerca del minimo con il minor costo teoricamente possibile, cioè $O(\ln(N^2))$. Questo limite è ottenibile solamente memorizzando i valori in un albero e utilizzando un algoritmo di tipo *heapsort*. Per ulteriori approfondimenti si veda [24].

2.5.4 Condizioni su ∂Q

Le condizioni sul bordo della griglia non necessitano di particolari attenzioni. Non è infatti necessario assegnare *a priori* un valore ai nodi al bordo.
È possibile aggiungere ai nodi della griglia una "cornice" di nodi fantasma ai quali si assegna inizialmente il valore $T = +\infty$. Quando essi entrano per la

[1] In Matlab è più esatto parlare di *array* dinamici.

prima volta nella *narrow band* e devono quindi essere calcolati, ad essi viene nuovamente assegnato d'ufficio il valore $T = +\infty$, in modo tale che non vengano mai utilizzati per il calcolo dei nodi successivi. Quando il valore più piccolo dei nodi della *narrow band* sarà proprio $+\infty$, i nodi fantasma saranno gli unici nodi ancora non *accepted* e si potrà quindi arrestare il calcolo.

Capitolo 3

Il metodo FMM-SL

Nel capitolo precedente abbiamo studiato il FMM, il cui pregio principale è il costo computazionale, di gran lunga inferiore a quello dei metodi basati sulla tecnica del punto fisso. D'altro canto, i metodi di approssimazione semilagrangiani (SL), descritti nei paragrafi 1.2 e 1.5.1, hanno il merito di essere più accurati e di non esigere una condizione CFL come la (2.10). Inoltre lo schema semilagrangiano è costruito in modo tale da sfruttare appieno i risultati teorici noti per l'equazione (2.1), rendendo più semplice la dimostrazione della convergenza. Infine, per questo schema sono state ottenute stime dell'errore in L^∞.
Appare quindi naturale cercare un nuovo metodo numerico che erediti i pregi del FMM e del SL.
Osserviamo che il tentativo di realizzazione di questo metodo è già presente in J. N. Tsitsiklis [29] e in J. A. Sethian, A. Vladimirski [27]. Questi ultimi lo realizzano nel caso (più generale) anisotropo, dove cioè

$$\mathbf{c} = c(x, n)n \qquad c : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}\,,$$

su una griglia non strutturata. In entrambi i casi però, la realizzazione è piuttosto ingenua. Di conseguenza il costo computazionale e la precisione di calcolo non sono più paragonabili con i due metodi originari.

In questo capitolo noi proponiamo una tecnica di realizzazione più raffinata, che ci permetterà di compiere un vero e proprio progresso sia rispetto allo schema FMM che allo schema SL.

3.1 Idee di base

Come abbiamo già accennato nella descrizione dei problemi di *tempo minimo*, la trasformata di Kruzkov

$$v(x) = 1 - e^{-T(x)}\,, \qquad T(x) = -\ln(1 - v(x))$$

permette di considerare il caso in cui il campo di velocità c si annulli in qualche punto. Questo è il caso, ad esempio, in cui nel dominio $Q\backslash\mathcal{T}$ siano presenti degli ostacoli che il fronte non può attraversare. Appare chiaro che se il punto dal quale si vuole raggiungere il *target* appartiene all'ostacolo, non esiste nessuna traiettoria soluzione del problema. Ai nodi che costituiscono l'ostacolo viene quindi assegnato il valore $v = 1$, corrispondente al valore $T = +\infty$.
Per poter ereditare questo vantaggio dal SL, anche nel FMM-SL useremo

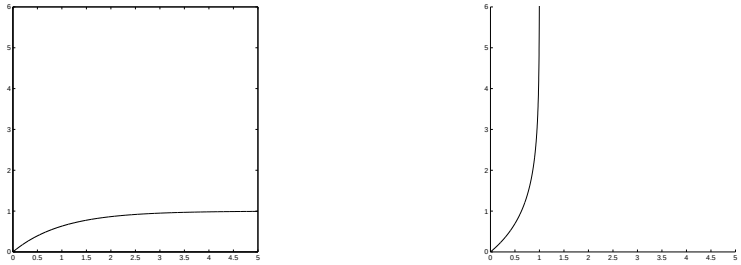


Figura 3.1: le funzioni v(T) (a sinistra) e T(v) (a destra)

questa trasformazione, approssimando la funzione v in luogo della funzione T. È fondamentale notare che la trasformazione di Kruzkov è monotòna crescente, cioè

$$T_1 > T_2 \Leftrightarrow v_1 > v_2 \tag{3.1}$$

come si può facilmente vedere dalla Figura 3.1. Ricordiamo infatti che il punto chiave del FMM sta nel convertire ad *accepted* il nodo della *narrow band* con il minimo valore per T, avanzando in questo modo "dal valore più piccolo al valore più grande". Grazie quindi alla (3.1) possiamo sostituire la funzione v alla funzione T senza dover rivedere le regole di avanzamento della *narrow band*.
L'idea di base del FMM-SL è semplice: si segue lo schema suggerito dal FMM fino al punto in cui deve essere calcolato il valore di un nodo. Esso non viene poi calcolato tramite un'approssimazione del gradiente bensì iterando fino a convergenza l'espressione

$$v_h(x_i) = \min_{a \in B(0,1)} \{\beta v_h(x_i - hc(x_i)a)\} + 1 - \beta. \tag{3.2}$$

In pratica, invece di applicare la procedura di punto fisso all'intera matrice dei valori dei nodi, la applichiamo "localmente" ad ogni singolo nodo, coprendo tutta la griglia secondo le regole del FMM.
Dimostreremo che, scegliendo un'approssimazione lineare per calcolare il valore di v_h nei punti che non coincidono con i nodi della griglia, il costo computazionale di questo nuovo metodo sarà comparabile con quello del FMM, in quanto sarà sufficiente una sola iterazione di punto fisso per arrivare a convergenza. Inoltre, la ricerca del *controllo ottimo* che realizza il minimo nella (3.2) potrà essere realizzata in modo molto rapido e preciso, abbandonando la tradizionale discretizzazione di $B(0,1)$.
Infine, nell'ultimo paragrafo, dimostreremo che anche un'approssimazione bilineare è compatibile con il FMM-SL, sebbene si perdano alcuni dei vantaggi appena descritti. Mostreremo invece che in questo approccio non possono essere accettate approssimazioni di ordine superiore, che coinvolgano più di quattro nodi.

3.2 Ricerca del controllo che realizza il minimo

Nei metodi semilagrangiani "classici" che abbiamo precedentemente studiato, la ricerca del *controllo ottimo* viene realizzata discretizzando la palla $B(1,0)$ in r punti, individuando i vettori

$$a_1, a_2, \ldots, a_k, \ldots, a_r \in B(1,0).$$

Questi vettori sono detti tipicamente *controlli*, in conformità alla terminologia del problema di *tempo minimo* (vedi Figura 3.2). Per ogni controllo a_k,

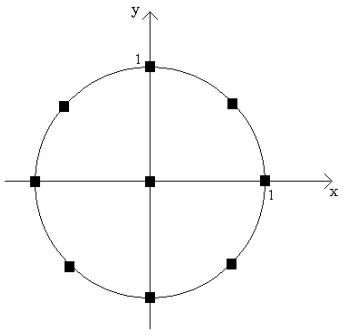


Figura 3.2: esempio di discretizzazione con 9 *controlli*

viene ricostruito il valore $v_h(x_i - hc(x_i)a_k)$ tramite una interpolazione di qualsiasi tipo e ordine. Le scelte di interpolazione più comuni sono:
lineare - tramite il valore di v_h nei tre nodi più vicini al punto $x_i - hc(x_i)a_k$.
bilineare - tramite il valore di v_h nei quattro vertici della cella in cui va a cadere il punto $x_i - hc(x_i)a_k$.
Una volta ottenuti i valori assunti dalla funzione v_h relativi agli r controlli, si procede ad un confronto per determinare il minore di essi.
Questo modo di procedere ha lo svantaggio di essere molto costoso computazionalmente (anche scegliendo un'interpolazione di ordine basso), ma ha il vantaggio di poter essere utilizzato con interpolazioni di ordine arbitrariamente alto.

È importante notare che il passo h scelto nella (3.2) non deve essere necessariamente uguale per ogni nodo. Da ora in poi indicheremo con h_i o con $h(x_i)$ il valore del passo h scelto per il nodo x_i.
Questo ci permette - nel caso in cui $c(x_i) > 0$ - di porre

$$h_i = \frac{\Delta x}{c(x_i)}. \tag{3.3}$$

Così facendo, la (3.2) diventa

$$v_h(x_i) = \min_{a \in B(0,1)} \{\beta v_h(x_i - \Delta x\ a)\} + 1 - \beta.$$

Introduciamo inoltre le seguenti notazioni:

$$c_i = c(x_i)\,, \qquad \beta_i = e^{-h_i}.$$

È importante notare che gli eventuali nodi in cui $c = 0$ possono essere trattati a parte, assegnando loro direttamente il valore $v = 1$ $(T = +\infty)$ senza effettuare nessun calcolo.

Il metodo di ricerca del *controllo ottimo* che proponiamo qui di seguito ha proprietà diverse dal metodo tradizionale: ha un basso costo computazionale ma è possibile utilizzarlo solo per un'approssimazione lineare. Inoltre la ricerca del minimo viene realizzata solamente su $\partial B(0,1)$ e non su tutta la palla come richiederebbe l'equazione (3.2). Quest'ultima restrizione comunque non rappresenta un problema. La scelta di una approssimazione lineare, unita alla condizione (3.3), garantisce che il minimo non possa essere raggiunto all'interno di $B(0,1)$.
In seguito mostreremo che questa tecnica può essere applicata con successo al FMM-SL.

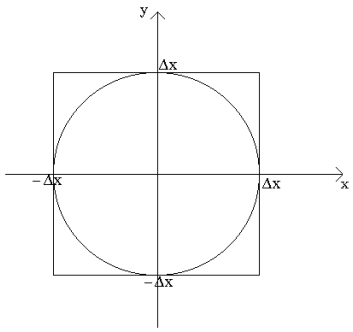


Figura 3.3: ricerca del controllo che realizza il minimo

Ragioniamo per semplicità su quattro celle di riferimento di lato Δx centrate nell'origine (vedi Figura 3.3). Il problema è cercare il minimo della funzione

$$v_h((0,0) - \Delta x\ a)\,, \quad \text{dove} \quad a = (\cos\theta, \sin\theta) \quad \text{e} \quad \theta \in [0, 2\pi)$$

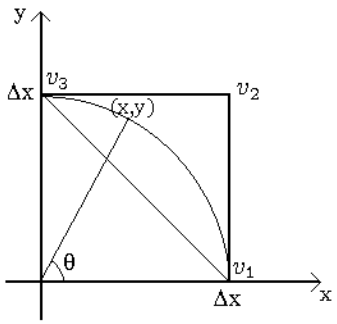
Definiamo a questo scopo un vettore $\mathbf{m} = (m_1, m_2, \ldots, m_8)$ assegnando un valore a ciascuna delle sue componenti così come spiegato qui di seguito. Come vedremo, il valore $p = \min\{m_1, m_2, \ldots, m_8\}$ sarà il valore cercato. Cominciamo con il porre

$$m_1 = v_h(\Delta x, 0)\,, \quad m_2 = v_h(0, \Delta x)\,, \quad m_3 = v_h(-\Delta x, 0)\,, \quad m_4 = v_h(0, -\Delta x).$$

Studiamo ora la funzione v_h all'interno di ogni singolo quadrante:

I quadrante

Siano v_1, v_2 e v_3 i valori che assume la funzione v_h rispettivamente nei nodi $(\Delta x, 0)$, $(\Delta x, \Delta x)$ e $(0, \Delta x)$.



È noto che esiste una e una sola funzione lineare $f(x, y)$ tale che

$$f(\Delta x, 0) = v_1\,, \quad f(\Delta x, \Delta x) = v_2\,, \quad f(0, \Delta x) = v_3.$$

La sua equazione è

$$f(x, y) = \left(\frac{\Delta x - y}{\Delta x}\right) v_1 + \left(\frac{x + y - \Delta x}{\Delta x}\right) v_2 + \left(\frac{\Delta x - x}{\Delta x}\right) v_3 \tag{3.4}$$

che può essere riscritta, riordinando i termini, nel seguente modo:

$$f(x, y) = ax + by + c$$

dove

$$a = \left(\frac{v_2 - v_3}{\Delta x}\right)\,, \qquad b = \left(\frac{v_2 - v_1}{\Delta x}\right)\,, \qquad c = v_1 - v_2 + v_3$$

Definiamo la funzione

$$F(\theta) = f(x, y)\,|_{\substack{x=\Delta x\cos\theta \\ y=\Delta x\sin\theta}} = a\Delta x\cos\theta + b\Delta x\sin\theta + c\,, \quad \theta \in [0, 2\pi)$$

L'idea è quella di ricostruire la funzione v_h nei punti

$$(\Delta x\cos\theta, \Delta x\sin\theta)\,, \quad \theta \in (0, \pi/2)$$

tramite la funzione lineare F. Siamo quindi interessati al valore minimo che la funzione $F(\theta)$ assume nell'intervallo $(0, \pi/2)$ (gli estremi $\theta = 0$ e $\theta = \pi/2$ sono già stati considerati nei valori m_1 e m_2). Derivando, otteniamo

$$F'(\theta) = -a\Delta x\sin\theta + b\Delta x\cos\theta$$

e, dividendo per $\cos(\theta)$ (sempre diverso da zero nell'intervallo considerato), otteniamo

$$F'(\theta) = 0 \Leftrightarrow \theta = \arctan(b/a).$$

È facile vedere che la funzione F ha un minimo relativo per $\theta \in (0, \pi/2)$ se e solo se $v_2 < v_1,\ v_3$. In caso contrario infatti, dal momento che la funzione f è lineare, il minimo valore di F (e di f) nel primo quadrante sarebbe v_1 oppure v_3 (quindi $\theta = 0$ oppure $\theta = \pi/2$).
Limitando quindi la nostra attenzione al caso

$$v_2 < v_1,\ v_3 \tag{3.5}$$

possiamo affermare che

$$a \neq 0\,, \quad b \neq 0\,, \quad b/a > 0\,, \quad \arctan(b/a) \in (0, \pi/2).$$

Di conseguenza, il punto

$$(x, y) = \Big(\Delta x \cos(\arctan(b/a)), \Delta x \sin(\arctan(b/a))\Big) \tag{3.6}$$

appartiene al primo quadrante e deve quindi necessariamente essere un punto di minimo relativo per la funzione $F(\theta)\,, \quad \theta \in [0, 2\pi)$. Inoltre notiamo che la funzione F ha solamente un altro punto critico (che è un massimo relativo) corrispondente al valore $\theta = \arctan(b/a) + \pi$ (vedi Figura 3.4). A

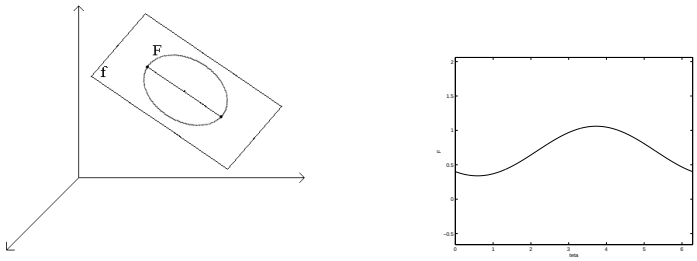


Figura 3.4: i due punti critici della funzione $F(\theta)$

questo punto non resta altro da fare che andare a sostituire nella (3.4) le coordinate (x, y) in (3.6) per ricostruire il valore approssimato di v_h in quel punto. Questo valore sarà m_5.

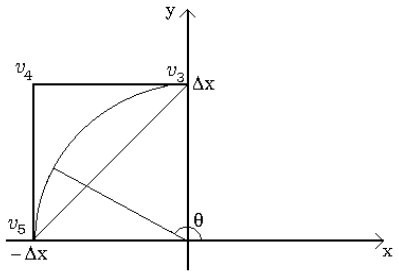
Se invece

$$v_2 \geq v_1,\ v_3$$

allora si pone $m_5 = +\infty$ (da intendersi come il massimo numero accettato dal calcolatore) dal momento che sicuramente il controllo che realizza il minimo non sarà all'interno del primo quadrante.

II quadrante

Siano v_3, v_4 e v_5 i valori che assume la funzione v_h rispettivamente nei nodi $(0, \Delta x)$, $(-\Delta x, \Delta x)$ e $(-\Delta x, 0)$.



L'unica funzione lineare $f(x, y)$ tale che

$$f(0, \Delta x) = v_3\,, \quad f(-\Delta x, \Delta x) = v_4\,, \quad f(-\Delta x, 0) = v_5$$

è

$$f(x, y) = \left(\frac{\Delta x + x}{\Delta x}\right) v_3 + \left(\frac{y - x - \Delta x}{\Delta x}\right) v_4 + \left(\frac{\Delta x - y}{\Delta x}\right) v_5$$

che può essere riscritta, riordinando i termini, nel seguente modo:

$$f(x, y) = ax + by + c$$

dove

$$a = \left(\frac{v_3 - v_4}{\Delta x}\right), \qquad b = \left(\frac{v_4 - v_5}{\Delta x}\right), \qquad c = v_3 - v_4 + v_5.$$

La funzione $F(\theta)$ ha un minimo relativo nell'intervallo $(\pi/2, \pi)$ se e solo se

$$v_4 < v_3,\ v_5. \tag{3.7}$$

In questo caso si ha che

$$a \neq 0\,, \quad b \neq 0\,, \quad b/a < 0\,, \quad \arctan(b/a) \in (-\pi/2, 0).$$

È importante ricordare che la funzione $\arctan(x)$ assume sempre valori nell'intervallo $(-\pi/2, \pi/2)$. Indicheremo invece con $\overline{\arctan}(x)$ la funzione (polidroma!) inversa di $\tan(x)$, $x \in \mathbb{R}$.
In questo caso il valore cercato non è $\theta = \arctan(b/a)$ bensì

$$\theta = \arctan(b/a) + \pi$$

cioè l'unico valore che la funzione $\overline{\arctan}(x)$ può assumere nell'intervallo $(\pi/2, \pi)$ (vedi Figura 3.5). Quest'ultimo valore deve necessariamente corrispondere ad un punto di minimo relativo per la funzione F.
A questo punto si procede come descritto nel caso del I quadrante assegnando

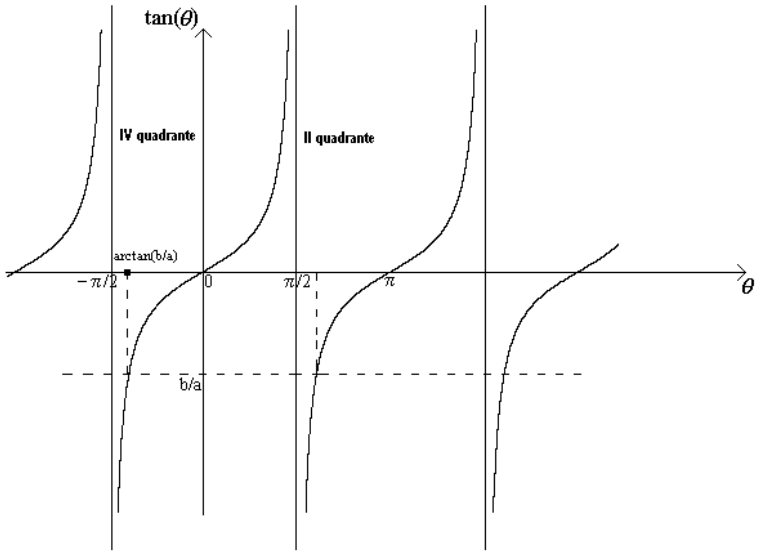
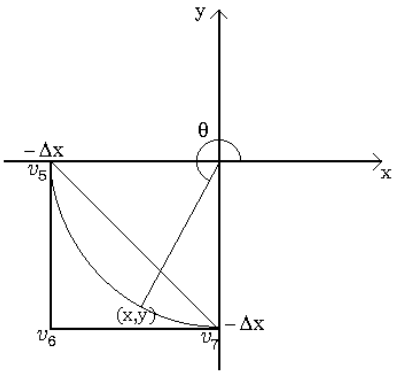


Figura 3.5: $\arctan(b/a)$

a m_6 il valore di f calcolato nel punto di minimo.
Nel caso in cui non valga la (3.7) invece, si pone $m_6 = +\infty$.

III quadrante

Siano v_5, v_6 e v_7 i valori che assume la funzione v_h rispettivamente nei nodi $(-\Delta x, 0)$, $(-\Delta x, -\Delta x)$ e $(0, -\Delta x)$.



Come nei casi precedenti avremo

$$f(x, y) = ax + by + c$$

dove

$$a = \left(\frac{v_7 - v_6}{\Delta x}\right), \qquad b = \left(\frac{v_5 - v_6}{\Delta x}\right), \qquad c = v_5 - v_6 + v_7.$$

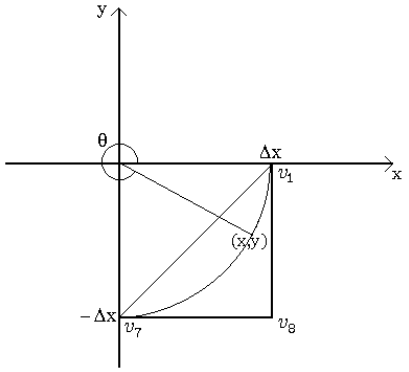
Nel caso in cui

$$v_6 < v_5,\ v_7$$

il valore di θ per cui la funzione f ha un minimo è $\theta = \arctan(b/a) + \pi$.

IV quadrante

Siano v_7, v_8 e v_1 i valori che assume la funzione v_h rispettivamente nei nodi $(0, -\Delta x)$, $(\Delta x, -\Delta x)$ e $(\Delta x, 0)$.



Come nei casi precedenti avremo

$$f(x, y) = ax + by + c$$

dove

$$a = \left(\frac{v_8 - v_7}{\Delta x}\right), \qquad b = \left(\frac{v_1 - v_8}{\Delta x}\right), \qquad c = v_7 - v_8 + v_1.$$

Nel caso in cui

$$v_8 < v_7,\ v_1$$

il valore di θ per cui la funzione f ha un minimo è $\theta = \arctan(b/a) + 2\pi$.

Assegnate tutte le componenti del vettore $\mathbf{m}$, non resta altro da fare che porre

$$p = \min\{m_1, m_2, \ldots, m_8\}$$

e calcolare l'espressione

$$v_h(0,0) = \beta p + 1 - \beta. \tag{3.8}$$

Si deve poi ripetere tutto il procedimento per ogni iterazione di punto fisso. Comunque, già adesso si può intuire che utilizzando una interpolazione lineare come quella appena descritta, la convergenza verrà raggiunta in una sola iterazione, dal momento che il cambiamento del valore v_h nel nodo che si sta considerando non ha alcun effetto nel calcolo della funzione F o f o nel valore v_h nei nodi adiacenti. Di conseguenza, ripetendo la ricerca del *controllo ottimo* ritroveremmo lo stesso risultato ottenuto precedentemente. Come vedremo la situazione sarà diversa con una diversa scelta dell'interpolazione.

A questo punto dovrebbe essere chiaro il vantaggio di questo metodo per la ricerca del *controllo ottimo* rispetto al metodo tradizionale di discretizzazione di $B(0,1)$. A fronte di un costo computazionale comparabile con quello che si avrebbe utilizzando 8 *controlli*, si ottiene il *controllo* esatto.

Concludiamo con un'osservazione riguardo all'errore dovuto all'aritmetica finita che si commette nel calcolo della (3.8). Supponiamo che nel dominio $Q\backslash\mathcal{T}$ sia presente un ostacolo, come descritto in Figura 3.6. Supponiamo in-

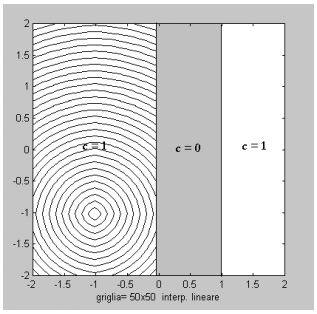


Figura 3.6: evoluzione di un fronte circolare in presenza di un ostacolo

oltre di voler ricostruire l'evoluzione di un fronte con configurazione iniziale puntiforme posto a sinistra dell'ostacolo. Sebbene a destra di esso la velocità sia positiva, il fronte non potrà mai raggiungere questa regione, dal momento che non può attraversare l'ostacolo. Nel momento in cui l'algoritmo calcola il valore dei nodi a destra dell'ostacolo, la situazione è la seguente:

$$c > 0\,, \quad h > 0\,, \quad \beta < 1\,, \quad p = 1$$

Di conseguenza la (3.8) (riscritta relativamente ad un generico nodo) diventa

$$v_h(x_i, y_j) = \beta + 1 - \beta. \tag{3.9}$$

Abbiamo osservato sperimentalmente che questa somma dà luogo ad un grande errore di arrotondamento. Per evitare questo problema è bene sostituire la (3.9) con

$$v_h(x_i, y_j) = \beta - \beta + 1.$$

Questo è un fatto noto in analisi numerica. È infatti sempre buona regola riordinare gli addendi secondo un ordine di valore assoluto non decrescente.

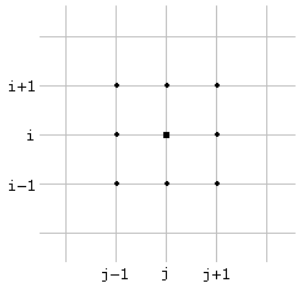
3.3 Descrizione dell'algoritmo

Introduciamo preliminarmente la definizione di insieme dei nodi *adiacenti*.

Definizione 3.1 *Sia* (x_i, y_j) *un generico nodo. L'insieme dei nodi* adiacenti *a questo nodo è:*

$$W = \Big\{(x_{i+1}, y_j), (x_{i-1}, y_j), (x_i, y_{j+1}), (x_i, y_{j-1}), (x_{i+1}, y_{j+1}), \\ (x_{i+1}, y_{j-1}), (x_{i-1}, y_{j+1}), (x_{i-1}, y_{j-1})\Big\}. \tag{3.10}$$

Anche questa definizione - così come la definizione di insieme dei *vicini* V di un nodo, data nel paragrafo 2.3 - può essere estesa ad una griglia di $\mathbb{R}^n$.



È facile vedere che

$$W = V \cup \Big\{(x_{i+1}, y_{j+1}), (x_{i+1}, y_{j-1}), (x_{i-1}, y_{j+1}), (x_{i-1}, y_{j-1})\Big\}.$$

Osserviamo inoltre che in questo caso non è possibile inizializzare tutti i nodi *far* con il valore $v = 1$ come sarebbe naturale pensando alla corrispondenza con il valore $T = +\infty$ del FMM. Infatti, dal momento che stiamo considerando la possibilità che la funzione $c(x)$ si possa annullare, nascerebbe

una confusione tra i nodi ai quali è stato assegnato il valore 1 perché ancora non sono stati calcolati e i nodi che invece hanno il valore 1 perché effettivamente non possono essere raggiunti dal fronte. Questo problema può essere risolto semplicemente assegnando $v = 2$ ai nodi *far*, o un qualsiasi altro valore maggiore di 1.
Passiamo ora alla descrizione dell'algoritmo.

Inizializzazione

1. Si individuano i nodi che costituiscono il fronte iniziale che vengono etichettati come *accepted*, assegnando loro il valore $v = 0$.

2. Si individuano i nodi della *narrow band* iniziale, cioè i nodi *adiacenti* al fronte, esterni se $c > 0$ o interni se $c < 0$. Essi vengono etichettati come *narrow band*, e si assegna loro il valore $v = 1 - e^{-\frac{\Delta x}{c}}$ (cioè $T = \Delta x/c$) se sono anche *vicini* di qualche nodo del fronte (come nel FMM) e il valore $v = 1 - e^{-\frac{\sqrt{2}\Delta x}{c}}$ (cioè $T = \sqrt{2}\Delta x/c$) altrimenti. Nella figura 3.7 è riportata la corretta inizializzazione nel caso di un fronte con configurazione iniziale puntiforme e velocità $c \equiv 1$.

3. Si etichettano come *far* tutti i rimanenti nodi della griglia assegnando loro il valore $v = 2$.

Nel capitolo dedicato al FMM è spiegato nel dettaglio come eseguire queste operazioni. In questo caso però, è necessario includere nella *narrow band* iniziale tutti i nodi *adiacenti* ai nodi che costituiscono il fronte, e non solamente i loro *vicini*.

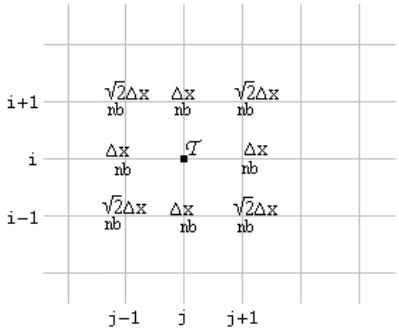


Figura 3.7: inizializzazione nel FMM-SL

Ciclo principale

1. Si individuano, tra i nodi della *narrow band*, quello cui è assegnato il minimo valore per v. Indicheremo questo nodo con A.

2. Si etichetta il nodo A con *accepted* e lo si rimuove dalla *narrow band*.

3. Si etichettano come *attivi* i nodi *adiacenti* ad A che non sono *accepted*. Se tra questi ci sono nodi *far*, essi vengono trasferiti nella *narrow band*.

4. Si calcola (o ricalcola) il valore v nei nodi che sono contemporaneamente *attivi* e *vicini* di A, iterando fino a convergenza l'espressione

$$v_h(x_i) = \min_{a \in B(0,1)} \{\beta v_h(x_i - h_i c_i a)\} + 1 - \beta \tag{3.11}$$

   Si calcola poi allo stesso modo il valore v nei rimanenti nodi *attivi*.

5. Se la *narrow band* non è vuota si ritorna al punto 1.

Nello schema appena descritto notiamo una sostanziale differenza con il FMM per quanto riguarda l'avanzamento della *narrow band*. Infatti, essa si propaga anche nelle direzioni "diagonali" a partire dai nodi che vengono considerati definitivi. Inoltre il calcolo dei nodi *adiacenti* non viene effettuato in un ordine casuale, ma dando la precedenza alle quattro direzioni N,S,E,W.
Per la gestione delle etichette *far*, *narrow band*, e *accepted*, per la ricerca del minimo valore di v e per le condizioni al bordo rimandiamo al FMM.

Velocità di propagazione infinita

Abbiamo visto nel paragrafo precedente che gli eventuali nodi in cui $c = 0$ possono essere calcolati a parte, assegnando loro direttamente il valore $v = 1$. Vogliamo ora mostrare che è altrettanto semplice trattare il caso in cui la velocità di propagazione del fronte sia infinita. Infatti, ragionando in maniera informale, supponiamo che esista un nodo x_{i_0} tale che

$$\lim_{x \to x_{i_0}} c(x) = +\infty$$

Definendo

$$g(x) = \frac{1}{c(x)}$$

l'equazione $c(x)|\nabla T(x)| = 1$ può essere riscritta nel seguente modo:

$$|\nabla T(x)| = g(x)$$

e si avrà

$$\lim_{x \to x_{i_0}} g(x) = 0.$$

Un'equazione di questo tipo prende il nome di *equazione eiconale degenere*, ed ha un grande interesse teorico e pratico. Nell'ultimo capitolo ne vedremo un'applicazione ai problemi di *shape from shading*.
Per calcolare $v_h(x_{i_0})$, possiamo porre, in accordo con la (3.3), $h_{i_0} = 0$ (e di conseguenza $\beta_{i_0} = 1$) e procedere poi al calcolo della (3.11) ponendo comunque

$$h_{i_0} c_{i_0} = \Delta x \tag{3.12}$$

Supponendo di estendere la funzione $h(x)$ a tutto il dominio $Q \backslash \mathcal{R}$ (e non solo ai punti corrispondenti ai nodi della griglia), la (3.12) può essere giustificata supponendo che si abbia

$$\lim_{x \to x_{i_0}} c(x) = +\infty\,, \quad \lim_{x \to x_{i_0}} h(x) = 0 \quad \text{e} \quad \lim_{x \to x_{i_0}} c(x)h(x) = \Delta x$$

Questo ragionamento, pur essendo non rigoroso, assegna al nodo x_{i_0} esattamente il valore v_h che ci si aspetta. Infatti, seguendo ancora la (3.11), otteniamo

$$v_h(x_{i_0}) = \min_{a \in B(0,1)} \{1 \cdot v_h(x_{i_0} - \Delta x a)\} + 1 - 1 = v_h(x_{i_0} - \Delta x a^*)$$

dove a^* è il *controllo ottimo*. Dal momento che la velocità di propagazione del fronte nel nodo x_{i_0} è infinita, il tempo di primo arrivo del fronte in quel punto è uguale al tempo di primo arrivo del fronte sulla circonferenza di raggio Δx centrata in x_{i_0}.

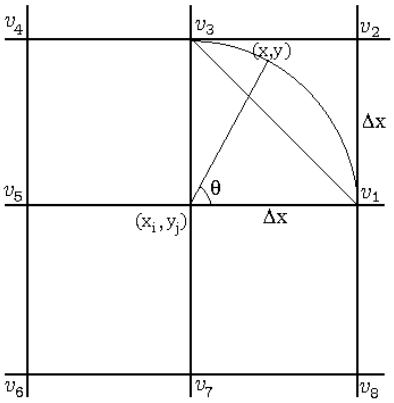
3.4 Dimostrazione della validità del metodo

In questo paragrafo mostreremo dapprima che iterare più di una volta il calcolo di $v_h(x_i)$ non porta ad una valore diverso per il nodo x_i. Analizzando i diversi casi che si possono incontrare, mostreremo come il metodo "veloce" per la ricerca del minimo descritto nel paragrafo 3.2 sia compatibile con il FMM-SL.
In seguito dimostreremo la convergenza del metodo alla soluzione di *viscosità* sfruttando i risultati ottenuti nella dimostrazione del FMM e i noti risultati a disposizione per il SL.

Supponiamo di dover calcolare il valore $v_h(x_i, y_j)$ e supponiamo, senza perdere di generalità, che

$$a^* = (\cos\theta,\ \sin\theta)\,, \quad \theta \in [0, \pi/2] \tag{3.13}$$

cioè che il minimo venga raggiunto nella cella "in alto a destra" rispetto al nodo (x_i, y_j), estremi inclusi. Con un abuso di notazione indicheremo questa cella riferendoci al "I quadrante".



Esaminiamo singolarmente tutti i casi possibili:

1. $v_1 = v_2 = v_3 = 2$.
   Questo caso non si può verificare. Infatti, dal momento che il minimo è raggiunto nel I quadrante, si deve avere

   $$v_4 = v_5 = v_6 = v_7 = v_8 = 2.$$

   D'altro canto sappiamo che se il nodo (x_i, y_j) deve essere calcolato, almeno uno dei suoi nodo *adiacenti* deve essere stato convertito ad *accepted*, e quindi avere un valore $v \in [0, 1]$.

2. Due valori tra v_1, v_2 e v_3 valgono 2.

   (a) $v_1 = v_3 = 2$: questo caso non si può verificare. Infatti, dal momento che il minimo è raggiunto nel I quadrante, deve essere $v_2 \leq v_1, v_3, v_4, \ldots, v_8$. Di conseguenza il nodo che viene convertito ad *accepted* è proprio v_2 (per semplicità confondiamo il nodo con il suo valore) e quindi si dovrà dare la precedenza al calcolo di v_1 e v_3.

   (b) $v_1 = v_2 = 2$: il valore minimo sarà v_3. Iterare nuovamente il calcolo di $v_h(x_i, y_j)$ non porterebbe ad un valore diverso dal precedente.

(c) $v_2 = v_3 = 2$: il minimo sarà v_1. Anche in questo caso la convergenza è raggiunta in una sola iterazione.

3. Un valore tra v_1, v_2 e v_3 vale 2

   (a) $v_2 = 2$: per la linearità della funzione f il minimo sarà v_1 o v_3. La convergenza è raggiunta in una sola iterazione.

   (b) $v_1 = 2\,, \quad v_3 \leq v_2$: il minimo sarà v_3.

   (c) $v_1 = 2\,, \quad v_3 > v_2$: questo è il caso più delicato, in cui $v_2 < v_1,\ v_3$. Si ha quindi un minimo in corrispondenza di un valore $\theta \in (0, \pi/2)$. Il valore del nodo (x_i, y_j), ricostruito tramite l'interpolazione lineare, è però "inquinato" dal valore $v_1 = 2$, che è un valore fittizio, molto diverso dal suo valore definitivo. Inoltre, iterare il calcolo non servirebbe, dal momento che il valore v_1 non può cambiare e che quindi il valore calcolato per il nodo (x_i, y_j) rimarrà lo stesso. Ci troviamo quindi in una situazione in cui la convergenza non viene raggiunta neanche dopo infinite iterazioni. Osserviamo che nello schema numerico SL questa situazione era esclusa dal momento che *tutti i nodi della griglia venivano calcolati contemporaneamente* e che nel FMM era esclusa perché i nodi con il valore $+\infty$ erano automaticamente lasciati fuori dal calcolo.
   Fortunatamente la seguente proposizione dimostra che è da escludersi il verificarsi di una situazione di questo genere.

   Proposizione 3.2 *Sia verificata l'ipotesi* (3.13). *Sia inoltre* $v_1 = 2$ *e* $v_3 > v_2$.
   Allora il nodo (x_i, y_j) *non sarà calcolato tramite un'interpolazione che utilizza valori fittizi della griglia.*

   Dimostrazione. Cominciamo con il dimostrare che il nodo (x_i, y_j) non deve essere calcolato perché uno dei suoi *vicini* è diventato *accepted*, cioè uno dei suoi *vicini* è nella *narrow band* ed ha il minimo valore per v tra tutti i nodi della *narrow band*.
   Il fatto che $v_1 = 2$ implica che nessun nodo *adiacente* a v_1 è ancora stato convertito ad *accepted*. Di conseguenza v_2 e v_3 sono nodi della *narrow band*. Inoltre, deve essere

   $$v_5,\ v_7 \geq v_2$$

   perché in caso contrario il minimo non sarebbe raggiunto nel primo quadrante.
   Abbiamo quindi dimostrato che v_2 è nella *narrow band* ed è minore del valore di tutti gli altri *vicini* del nodo (x_i, y_j).

Di conseguenza, se il nodo (x_i, y_j) deve essere calcolato, è perché uno tra i nodi v_2, v_4, v_6 o v_8 è diventato *accepted*. Studiamo singolarmente questo quattro casi:
- v_2: per come è stato descritto l'algoritmo, verrà calcolato prima v_1 e solo dopo il nodo (x_i, y_j).
- v_8: analogo al caso precedente.
- v_4: sebbene si abbia $v_4 \leq v_2$, il valore $v_h(x_i, y_j)$ sarà maggiore di v_2, in quanto esso viene calcolato tramite una combinazione convessa dei valori v_1, v_2 e v_3. Di conseguenza, il nodo (x_i, y_j) non sarà convertito ad *accepted* prima di v_2. Quando v_2 diventerà *accepted* saranno calcolati prima i valori v_1 e v_3 e solo dopo sarà ricalcolato il nodo (x_i, y_j), questa volta senza che entrino in gioco valori fittizi.
Se tra la conversione ad *accepted* di v_4 e quella di v_2 dovesse essere convertito ad *accepted* qualche altro nodo il cui valore è calcolato tramite il valore "inquinato" di (x_i, y_j), esso sarà sicuramente ricalcolato dopo la conversione ad *accepted* di v_2.
- v_6: analogo al caso precedente. ■

(d) $v_3 = 2\,, \quad v_1 \leq v_2$: il minimo sarà v_1. La convergenza è raggiunta in una sola iterazione.

(e) $v_3 = 2\,, \quad v_1 > v_2$: Analogo al caso 3 (c)

4. Nessun valore tra v_1, v_2 e v_3 vale 2.
In questo caso il valore del nodo (x_i, y_j) viene ricostruito attraverso nodi già calcolati. Questo valore non può essere quindi "inquinato" da valori fittizi. Inoltre come nei casi precedenti la convergenza è raggiunta in una sola iterazione.

Osservazione 3.3 *Il caso* 3 (c) *e i casi ad esso riconducibili possono essere trattati in modo molto più semplice scartando d'ufficio il valore v_1 e calcolando il valore nel nodo (x_i, y_j) utilizzando solamente i valori v_2 e v_3. Così facendo l'errore commesso non sarebbe significativamente maggiore rispetto al metodo precedentemente descritto.*

Come abbiamo fatto per il FMM, dobbiamo ora dimostrare che il valore minimo dei nodi della *narrow band* è sicuramente esatto, ed è quindi lecito considerarlo definitivo. Anche qui considereremo esatto (o non migliorabile) quel valore che sicuramente non potrà essere sostituito, ad un passo successivo del procedimento, con un valore strettamente minore. Non stiamo ovviamente prendendo in considerazione gli errori legati all'approssimazione numerica.

Osservazione 3.4 *Se aggiornassimo la* narrow band *come nel FMM* (*calcolando cioè solo i* vicini *non* accepted *dei nodi appena convertiti ad* accepted) *potremmo facilmente mostrare degli esempi in cui il valore minimo della* narrow band *non è esatto.*

Per ogni nodo X della griglia, si ha:

$$\beta v_h(X - h_i c_i a^*) + 1 - \beta \geq v_h(X - h_i c_i a^*) \Leftrightarrow$$

$$(\beta - 1) v_h(X - h_i c_i a^*) \geq \beta - 1 \Leftrightarrow$$

$$v_h(X - h_i c_i a^*) \leq 1$$

Dal momento che v_h è sempre minore o uguale di 1, abbiamo dimostrato che

$$v_h(X) \geq v_h(X - h_i c_i a^*).$$

Inoltre, supponiamo che $v_h(X - h_i c_i a^*)$ sia stato calcolato utilizzando i valori $v_h^{(1)}$, $v_h^{(2)}$ e $v_h^{(3)}$, e che

$$v_h^{(1)} = \min \left\{ v_h^{(1)}, v_h^{(2)}, v_h^{(3)} \right\}.$$

Allora l'interpolazione lineare adottata assicura che

$$v_h(X) \geq v_h^{(1)} \tag{3.14}$$

In analogia alla proposizione 2.2 del paragrafo 2.4, enunciamo ora la seguente

Proposizione 3.5 *Sia $v_h(X)$ il valore che viene assegnato al nodo $X = (x_i, y_j)$ nel momento in cui il nodo A ad esso* adiacente *viene convertito ad* accepted. *Supponiamo inoltre che*

$$c(x) \geq 0\,, \quad \forall\ x \in Q \backslash \Omega.$$

Allora si avrà

$$v_h(X) \geq v_h(A)$$

Diamo alcune idee di base che potranno essere utili per lo sviluppo in una dimostrazione rigorosa della precedente proposizione.

Ragioniamo ancora per semplicità su quattro celle di riferimento centrate nell'origine e supponiamo che il minimo venga raggiunto nel I quadrante (vedi Figura 3.8).

CASO 1

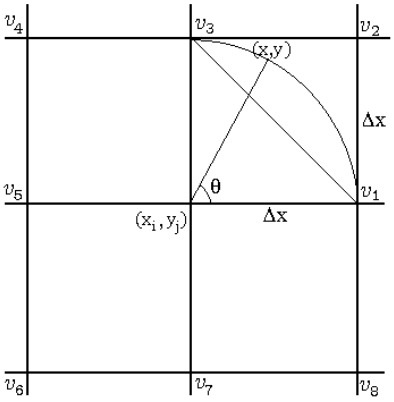


Figura 3.8: dimostrazione proposizione 3.5

$v_1, \ldots, v_8$ sono *narrow band* o *far*.
Se A appartiene al I quadrante, la (3.14) assicura la tesi. Altrimenti si avrà

$$v_h(X) \geq \min\left\{v_1, v_2, v_3\right\} \stackrel{def}{=} v^*$$

e $v^* \geq v_h(A)$ dal momento che A è il primo nodo ad essere stato convertito ad *accepted*.

CASO 2
Un nodo tra $v_1, \ldots, v_8$ è *accepted*.
Se il nodo *accepted* non appartiene al I quadrante, si ricade nel *CASO* 1.
Supponiamo ora che il nodo *accepted* appartenga al I quadrante. Lo indicheremo con P. Per fissare le idee, supponiamo che il nodo P sia v_2. Di conseguenza, il nodo A è v_1 o v_3. Il valore che viene assegnato al nodo X è, nella stragrande maggioranza dei casi, identico al valore $v_{old}(X)$ che gli era stato assegnato quando P è diventato *accepted*. D'altro canto $v_{old}(X) \geq v_h(A)$, dal momento che è stato scelto il nodo A per essere convertito ad *accepted*.
Nel caso in cui, nel tempo che intercorre tra la conversione ad *accepted* di P e quella di A, alcuni valori dell a *narrow band* che saranno coinvolti nel calcolo di $v_h(X)$ fossero sostituiti con dei valori minori, il valore assegnato ad X potrebbe effettivamente essere minore di $v_{old}(X)$, ma la scelta dell'approssimazione lineare dovrebbe comunque assicurare che $v_h(X) \geq v_h(A)$.

CASO 3
Più nodi tra $v_1, \ldots, v_8$ sono *accepted*.
In questo caso la situazione è riconducibile al *CASO* 2. ∎

Una volta acquisito questo risultato, la conclusione segue in modo perfettamente analogo al FMM.

Osserviamo quindi che da una parte sfruttiamo i risultati noti per lo schema SL, i quali assicurano che il valore calcolato nel nodo X tramite la (3.11) sia effettivamente un valore approssimato di $v(X)$ (in questo caso la convergenza è sempre raggiunta in una sola iterazione ma è importante osservare che la funzione $f(x) = \beta x + 1 - \beta$ è effettivamente una contrazione). Dall'altra parte, sfruttiamo il modo di avanzamento della *narrow band* per dimostrare che il valore che viene convertito ad *accepted* non può essere migliorato, anche se da quel momento in poi calcolassimo contemporaneamente tutti i valori della griglia come nello schema SL.
Grazie ai risultati generali del teorema 2.3, possiamo quindi concludere che il FMM-SL fornisce esattamente la stessa soluzione dello schema SL, ereditandone tutti i risultati teorici. I test numerici confermano l'uguaglianza delle soluzioni numeriche fornite dai due metodi.

3.5 Approssimazioni di ordine superiore

Nel caso di una approssimazione bilineare, il valore v_h viene ricostruito tramite il valore di quattro nodi. Ragionando su quattro celle di riferimento centrate nell'origine e scegliendo ancora il passo h in modo tale che

$$c_i h_i = \Delta x \quad \text{per ogni } i$$

il valore $v_h(x_i - c_i h_i a)$ viene ricostruito tramite i quattro vertici della cella in cui cade il punto $x_i - c_i h_i a$. Nella Figura 3.9 è mostrato a titolo di esempio il caso in cui il *controllo ottimo* è raggiunto nel I quadrante e il valore $v_h((0,0) - c_i h_i a)$ viene ricostruito tramite i valori v_1, v_2, v_3 e $v_h(0,0)$. La funzione bilineare f tramite la quale si vuole ricostruire il valore della funzione v_h è della forma

$$f(x,y) = axy + bx + cy + d$$

dove a, b, c e d sono coefficienti da determinare. La matrice hessiana della funzione f è

$$H(x,y) = \begin{pmatrix} 0 & a \\ a & 0 \end{pmatrix}$$

e quindi $\det H(x,y) = -a^2 \leq 0$. Se $a = 0$ la funzione f è ovviamente lineare, mentre se $a \neq 0$ la funzione f ha un unico punto critico, corrispondente ad

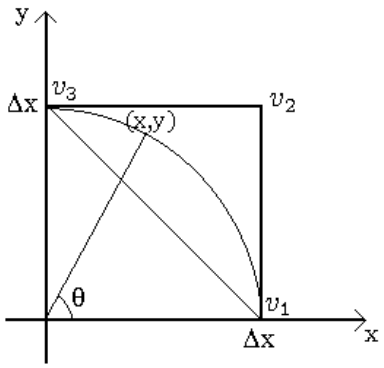


Figura 3.9: minimo raggiunto nel I quadrante

un punto di sella, in $(-c/a, -b/a)$. In ogni caso quindi la funzione f non ha nessun punto di minimo relativo. Questo proprietà è fondamentale per poter dimostrare la validità del FMM-SL con questo tipo di approssimazione. Infatti, così come nel caso dell'interpolazione lineare, se il valore $v_h(X - h_i c_i a^*)$ è stato calcolato utilizzando i valori $v_h^{(1)}$, $v_h^{(2)}$, $v_h^{(3)}$ e $v_h^{(4)}$, e se

$$v_h^{(1)} = \min\left\{v_h^{(1)}, v_h^{(2)}, v_h^{(3)}, v_h^{(4)}\right\}$$

allora si avrà sicuramente

$$v_h(X) \geq v_h^{(1)}.$$

Grazie a questa proprietà è possibile ottenere gli stessi risultati che si hanno per l'approssimazione lineare nei casi 1, 2, 3 e 4 studiati precedentemente. L'unica differenza la incontriamo nel caso 4, in cui la convergenza non è più raggiunta in una sola iterazione di punto fisso. Infatti, nella prima iterazione il valore $v_h((0,0) - c_i h_i a)$ viene ricostruito anche tramite il valore $v(0,0) = 2$ (valore fittizio, lontano dal valore esatto). Di conseguenza il primo valore che viene assegnato al nodo $(0,0)$ sarà molto diverso dal suo valore esatto. D'altro canto, il cambiamento del valore $v_h(0,0)$ influisce sul calcolo di $v_h(0,0)$ stesso all'iterazione successiva. Così come abbiamo visto nei metodi semilagrangiani, quindi, iterando il procedimento, il valore $v_h(0,0)$ convergerà monotonamente verso un certo valore compreso nell'intervallo $(0,1)$ che sarà da considerarsi il valore esatto (nel senso di "non migliorabile").
Si potrebbe sperare che, sebbene siano necessarie *a priori* infinite iterazioni di punto fisso per arrivare a convergenza, la ricerca del *controllo ottimo* possa essere compiuta una volta sola. In altre parole, si potrebbe pensare che sia

sufficiente iterare l'espressione

$$v_h(0,0) = \beta v_h((0,0) - \Delta x a^*) + 1 - \beta$$

invece dell'espressione

$$v_h(0,0) = \min_{a \in B(0,1)} \Big\{ \beta v_h((0,0) - \Delta x a) \Big\} + 1 - \beta$$

Purtroppo anche sperimentalmente si osserva che ciò non è possibile, dal momento che il *controllo ottimo* a^* cambia al diminuire di $v_h(0,0)$.

È anche importante notare che, utilizzando un'approssimazione bilineare, si deve necessariamente tornare alla discretizzazione della palla $B(0,1)$ perché non è più possibile trovare analiticamente il *controllo ottimo* come nel caso dell'approssimazione lineare. Infatti, se definiamo

$$F(\theta) = f(x,y) \mid_{\substack{x=\Delta x \cos\theta \\ y=\Delta x \sin\theta}} = a(\Delta x)^2 \cos\theta \sin\theta + b\Delta x \cos\theta + \\ + c\Delta x \sin\theta + d\,, \quad \theta \in [0, 2\pi) \tag{3.15}$$

si ha che

$$F'(\theta) = a\Delta x(\cos^2\theta - \sin^2\theta) - b\sin\theta + c\cos\theta.$$

ma, sfortunatamente, l'equazione $F'(\theta) = 0$ non può essere risolta esplicitamente.

Concludiamo infine accennando all'utilizzo di approssimazioni di ordine superiore. È facile vedere che nel caso in cui si interpoli il valore di un generico punto x tramite 5 o più valori dei nodi, il caso 3 (c) diventa critico. Infatti non è più possibile escludere *a priori* la possibilità che entri in gioco nell'interpolazione un nodo ancora non calcolato (quindi con valore $v_h = 2$) diverso dal nodo che si sta calcolando. Di conseguenza il valore che viene calcolato tramite la (3.11) è affetto da un grande errore, che non può diminuire ripetendo il calcolo.

Capitolo 4

Evoluzioni del metodo FMM-SL

In questo capitolo descriveremo alcune varianti del metodo FMM-SL. Dapprima considereremo uno schema numerico molto simile al FMM-SL, che utilizza una doppia *narrow band* e calcola la soluzione con un costo computazionale decisamente minore. In seguito descriveremo alcune idee per la realizzazione di un nuovo metodo numerico, basato su una diversa concezione di *narrow band*, che dovrebbe adattarsi anche a casi più generali dell'equazione (2.1).

4.1 Il metodo FMM-SL con doppia *narrow band*

Come abbiamo già osservato nel paragrafo 2.3.1, nel FMM ogni nodo viene calcolato da 1 a 4 volte. Allo stesso modo, è facile vedere che nel FMM-SL ogni nodo viene calcolato da 1 a 8 volte, dal momento che esso può essere *adiacente* ad un nodo che viene convertito ad *accepted* al massimo per 8 volte.
Se è vero che in alcuni casi è effettivamente necessario ricalcolare il valore di un nodo, nella maggioranza dei casi fare questo è assolutamente inutile. Si possono infatti facilmente mostrare numerosi esempi dove il nuovo valore calcolato è esattamente identico al precedente. Nasce quindi l'esigenza di un miglioramento nella gestione della *narrow band* teso ad eliminare calcoli inutili. Il miglioramento che proponiamo in questo capitolo consiste nel calcolare (quando è possibile farlo) solamente i nodi *far* che sono *adiacenti* ai nodi appena diventati *accepted*, senza ricalcolare anche i nodi *adiacenti* che sono già nella *narrow band* (che quindi sono già stati calcolati almeno una volta). Quando invece si riconosce una situazione in cui è necessario rical-

colare un nodo già calcolato per assegnargli un valore diverso da quello che ha, si torna al vecchio (e *più lento*) modo di avanzamento della *narrow band*, calcolando cioè sia i nodi *adiacenti far* che quelli già nella *narrow band*.
Così facendo, l'avanzamento della *narrow band* risulta più veloce, pur continuando, come vedremo, a seguire correttamente il profilo del fronte mentre evolve nel tempo.
In questo capitolo descriveremo tale metodo pensandolo come un miglioramento del FMM-SL, ma è importante notare che può essere pensato anche come miglioramento del FMM.

4.1.1 Descrizione dell'algoritmo

L'inizializzazione procede come per il FMM-SL, dove vengono individuati i nodi facenti parte del fronte iniziale (assegnando loro il valore 0) e la *narrow band* iniziale costituita dai nodi *adiacenti* al fronte.
In questo metodo esiste anche una seconda *narrow band* che all'inizio del procedimento è vuota.
Descriviamo ora in maniera sintetica l'algoritmo per poi commentarlo più dettagliatamente:

(A) Se la seconda *narrow band* è vuota vai al ciclo 1, altrimenti vai al ciclo 2.

Ciclo 1 (prima *narrow band)*

1. Individuare tra i nodi della prima *narrow band* quello cui è assegnato il minimo valore per v.
2. Etichettare questo nodo con *accepted* e rimuoverlo dalla prima *narrow band*.
3. Individuare nuovamente tra i nodi della prima *narrow band* quello cui è assegnato il minimo valore per v. Sia x_{min} questo nodo e sia v_{min} il suo valore.
4. Etichettare come *attivi* i nodi *adiacenti* a x_{min} che sono *far*.
5. Calcolare il valore di v nei nodi *attivi* risolvendo l'equazione

$$v_h(x_i) = \min_{a \in B(0,1)} \left\{ \beta v_h(x_i - c(x_i) h_i a) \right\} + 1 - \beta \tag{4.1}$$

in modo analogo al FMM-SL, dando precedenza ai *vicini*.

6. Se a uno di questi nodi *attivi* viene assegnato un valore minore di v_{min}, memorizzarlo nella seconda *narrow band*, altrimenti memorizzarlo nella prima *narrow band*.

7. Torna a (A).

Ciclo 2 (seconda *narrow band)*

1. Individuare tra i nodi della seconda *narrow band* quello cui è assegnato il minimo valore per v.

2. Etichettare questo nodo con *accepted* e rimuoverlo dalla seconda *narrow band*.

3. Etichettare come *attivi* i nodi ad esso *adiacenti* che non sono *accepted* (cioè che sono *far* o che stanno nella prima o nella seconda *narrow band*).

4. Calcolare il valore di v nei nodi *attivi* risolvendo l'equazione (4.1) in modo analogo al FMM-SL.

5. Se a uno di questi nodi *attivi* viene assegnato un valore minore di v_{min}, memorizzarlo nella seconda *narrow band*, altrimenti memorizzarlo nella prima *narrow band*.

6. Torna a (A).

Soffermiamoci sulle differenze tra il metodo appena descritto e il FMM-SL. Si nota immediatamente che il *ciclo* 2 è identico al procedimento del FMM-SL, dove vengono calcolati tutti i nodi *adiacenti* non *accepted* dei nodi appena diventati *accepted*. Nel *ciclo* 1 invece troviamo la differenza principale. Vengono infatti calcolati solamente i nodi *adiacenti far* dei nodi appena diventati *accepted*. Così facendo nessun nodo viene calcolato più di una volta, ed è questo il motivo per cui si può affermare che l'avanzamento della prima *narrow band* è più veloce dell'avanzamento della seconda *narrow band*.
Analizziamo ora il meccanismo che regola la creazione e l'eliminazione delle due *narrow band*. Per fissare le idee ragioniamo su un esempio: supponiamo di avere un fronte con configurazione iniziale puntiforme (coincidente con un nodo della griglia) che evolve a velocità costante pari ad 1. In questo caso il fronte si propaga in maniera uniforme lungo tutte le direzioni. La situazione è illustrata nella Figura 4.1 dove è indicato in ogni nodo il tempo T di primo arrivo. Ricordiamo per comodità la trasformata di Kruzkov, data dalle relazioni

$$T = -\ln(1-v)\,, \qquad v = 1 - e^{-T}.$$

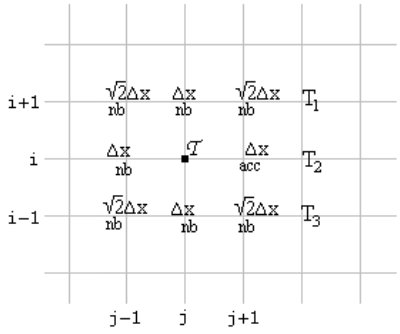


Figura 4.1: la seconda *narrow band* rimane vuota

Come è noto, l'inizializzazione consiste nell'assegnare il valore $v = 0$ (cioè $T = 0$) al nodo che costituisce il fronte, il valore $v = 1 - e^{-\Delta x}$ (cioè $T = \Delta x$) ai suoi quattro *vicini* e il valore $v = 1 - e^{-\sqrt{2}\Delta x}$ (cioè $T = \sqrt{2}\Delta x$) ai quattro nodi posti sulle diagonali. Questi otto nodi costituiscono la prima *narrow band*.
All'inizio del procedimento l'algoritmo entra nel *ciclo* 1 e passa ad *accepted* un nodo qualsiasi della *narrow band* con il valore $T = \Delta x$. Sia esso (x_i, y_{j+1}). Si pone inoltre $v_{min} = \Delta x$. L'algoritmo calcola poi i 3 nodi *adiacenti far* del nodo appena diventato *accepted* senza ricalcolare i nodi (x_{i-1}, y_{j+1}) e (x_{i+1}, y_{j+1}), ai quali sarebbe stato assegnato esattamente lo stesso valore. Appare chiaro che i valori T_1, T_2 e T_3 sono maggiori di v_{min} e non si assiste quindi alla nascita della seconda *narrow band*. Finché la velocità di propagazione del fronte rimane uniforme in tutte le direzioni lo schema si comporta in modo simile a quanto descritto e si procede quindi iterando solamente il *ciclo* 1.
Supponiamo ora che siano stati convertiti ad *accepted* i nodi (x_i, y_{j+1}), (x_{i-1}, y_j) e (x_{i+1}, y_j) (vedi Figura 4.2). Viene quindi convertito ad *accepted* il nodo (x_i, y_{j-1}) e si pone $v_{min} = \sqrt{2}\Delta x$. Supponiamo inoltre che $c(x_i, y_{j-2})$ sia maggiore di 1. Di conseguenza, è possibile che si abbia

$$v_h(x_i, y_{j-2}) = q\Delta x \quad \text{con} \quad 0 < q < \sqrt{2}.$$

In questo caso viene creata la seconda *narrow band*. Il *ciclo* 1 (e con esso la prima *narrow band*) si congela e si entra nel *ciclo* 2. Questo vuol dire fisicamente che il fronte sta cessando di evolversi uniformemente in tutte le direzioni e sta nascendo una sorta di *protuberanza*. Dal punto di vista controllistico questo corrisponde all'aver trovato un percorso veloce con conseguente abbassamento dei tempi di primo arrivo in quella direzione. (vedi

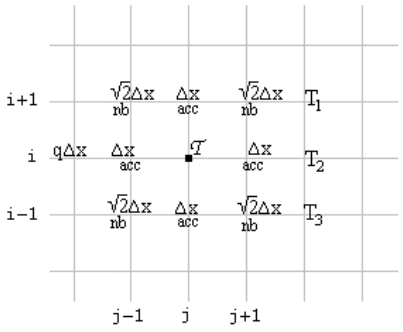


Figura 4.2: nascita della seconda *narrow band*

Figura 4.3)
La seconda *narrow band* si svuoterà completamente solo quando saranno stati calcolati tutti i nodi coinvolti nella *protuberanza*, cioè fino a quando si sarà ristabilito l'equilibrio tra i valori dei nodi su tutto il perimetro del fronte. A questo proposito è importante far notare che il valore v_{min}, memorizzato l'ultima volta che si era entrati nel *ciclo* 1, non viene mai sostituito all'interno del *ciclo* 2. È grazie ad esso che all'interno del *ciclo* 2 si può riconoscere quando è stato ristabilito nuovamente l'equilibrio.
Una volta dissolta la seconda *narrow band*, lo schema riparte iterando solamente il *ciclo* 1.
Come sarà dimostrato nel prossimo paragrafo, la situazione "pericolosa" nella quale è possibile che un nodo debba effettivamente essere ricalcolato, è proprio la nascita di questa *protuberanza*. Se questa *protuberanza* è tale da ripiegarsi su sé stessa (vedi Figura 4.4) vuol dire che dopo aver assegnato a un nodo un tempo di primo arrivo T_1 si riesce a tornare in quel nodo attraverso una strada diversa, più lunga ma più veloce, in un tempo $T_2 < T_1$.

Precisazioni sull'implementazione

1. È importante notare che nel punto 1 del *ciclo* 1 non è necessario fare una vera ricerca del minimo valore di v in quanto questo valore coincide con il valore v_{min} calcolato l'ultima volta che è stato eseguito il *ciclo* 1. Di conseguenza il costo computazionale complessivo per la ricerca del minimo valore dei nodi delle *narrow band* è identico a quello del FMM-SL e del FMM.

2. Nel punto 5 del *ciclo* 2 si deve fare attenzione all'eventuale passaggio di un nodo da una *narrow band* all'altra. Supponiamo infatti di aver

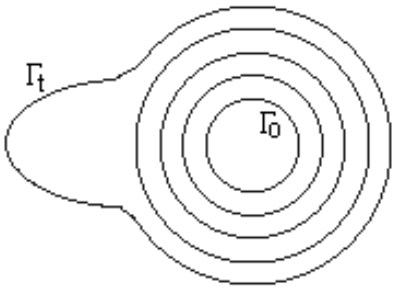


Figura 4.3: fronte che si evolve in modo irregolare

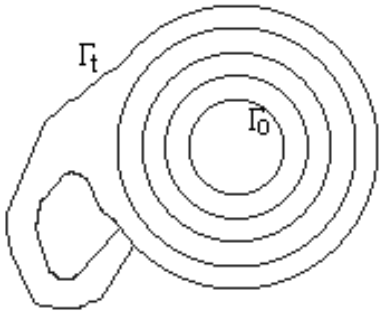


Figura 4.4: protuberanza che si ripiega su sé stessa

appena calcolato il valore del nodo X e di avergli assegnato il valore $v(X)$.
Se X è *far* si procede come descritto.
Se X sta nella prima *narrow band* e $v(X) < v_{min}$ allora va cancellato dalla prima e aggiunto alla seconda *narrow band*. Inoltre è necessario ricercare il nodo con il minimo valore di v nella prima *narrow band* (potrei averlo perso cancellando X).
Se X sta nella prima *narrow band* e $v(X) \geq v_{min}$ allora esso va lasciato nella prima *narrow band*.
Infine, se X sta nella seconda *narrow band*, non c'è possibilità che esso debba passare alla prima *narrow band* (vorrebbe dire dover correggere il valore di X con un valore più alto di $v(X)$). Esso va quindi lasciato nella seconda *narrow band*.

3. Lo schema numerico termina quando entrambe le *narrow band* sono vuote.

Per la gestione delle condizioni al bordo, dell'inizializzazione, della ricerca del minimo valore di v e delle etichette *far*, *accepted* e *narrow band*, rimandiamo alla descrizione del FMM e del FMM-SL.

4.1.2 Dimostrazione della validità del metodo

Come abbiamo precedentemente osservato, se la seconda *narrow band* non è vuota (cioè ci si trova all'interno del *ciclo* 2), ritroviamo esattamente il procedimento del FMM-SL, di cui supponiamo provata la validità. Tenendo conto di ciò, per dimostrare la validità di questo metodo è sufficiente verificare la seguente proposizione.

Proposizione 4.1 *Supponiamo che la seconda* narrow band *sia vuota. Supponiamo inoltre che nella prima* narrow band *ci sia il nodo X al quale è stato assegnato il valore $v(X)$ l'ultima volta che è stato calcolato. Supponiamo infine che esso venga ricalcolato e gli venga assegnato il valore $v_{new}(X)$.*
Allora, si avrà

$$v_{new}(X) = v(X)$$

Dimostrazione. Supponiamo che il nodo X sia stato calcolato l'ultima volta quando il nodo Z (con valore $v(Z)$) *adiacente* ad X è diventato *accepted* (vedi Figura 4.5). Sia inoltre X_{min} il nodo con il valore minimo tra tutti i nodi della prima *narrow band* a questo punto del procedimento. Indicheremo con v_{min} questo valore.
Procederemo per assurdo. Supponiamo ora che al nodo X debba essere assegnato un valore diverso da $v(X)$. Questo vuol dire che gli deve essere assegnato un valore strettamente minore di quello che ha (sicuramente non maggiore perché il valore esatto è il tempo di *primo* arrivo!).
Quindi necessariamente ci dobbiamo trovare nella situazione in cui, dopo l'assegnazione a X del valore $v(X)$, si è aperto un nuovo sentiero che mi porta al nodo X in un tempo minore di $T(X) = -\ln(1 - v(X))$.

Il punto chiave è che la nascita di questo sentiero (la *protuberanza* di cui sopra) deve essere annunciata dall'assegnazione a un qualche nodo di un valore minore di v_{min} (e quindi dalla nascita della seconda *narrow band*).

Infatti, supponiamo di procedere con l'avanzamento del fronte e di non assegnare mai ai nodi incontrati un valore minore di v_{min}. Dal momento che X deve essere ricalcolato, si dovrà prima o poi raggiungere un nodo *adiacente* ad esso, ad esempio il nodo Y. Quindi, avendo supposto di non assegnare

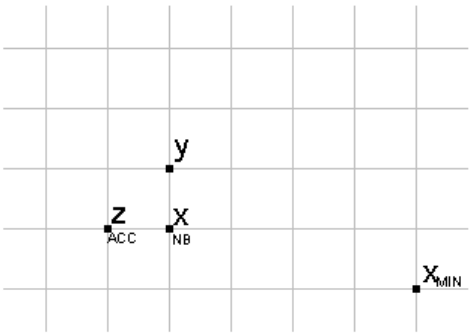


Figura 4.5: dimostrazione della proposizione 4.1

mai un valore minore di v_{min}, dovrà essere

$$v(Y) \geq v_{min}$$

D'altro canto sappiamo che $v(Z) \leq v_{min}$ perché il nodo Z è diventato *accepted* prima del nodo X_{min}. Quindi si avrà

$$v(Y) \geq v_{min} \geq v(Z) \tag{4.2}$$

Procedendo con lo schema numerico, il nodo Y diventerà *accepted* e verrà quindi calcolato $v_{new}(X)$ ma la (4.2) assicura che

$$v_{new}(X) \geq v(X).$$

Abbiamo quindi dimostrato che se voglio assegnare ad X un valore minore di $v(X)$, devo per forza assegnare precedentemente a qualche nodo un valore minore di v_{min} e quindi far nascere la seconda *narrow band*. ■

Abbiamo dimostrato che se la seconda *narrow band* è vuota, non è necessario ricalcolare i valori dei nodi della prima *narrow band*. Questa affermazione potrebbe ingannare il lettore disattento inducendolo a pensare che sia possibile passare ad *accepted* tutti questi nodi contemporaneamente e aggiornare in un solo colpo tutta la prima *narrow band*. Ma questo sarebbe sbagliato perché mentre vengono passati (uno alla volta) ad *accepted* i nodi della prima *narrow band*, potrebbe nascere una *protuberanza* che obbliga a ricalcolare uno dei nodi della prima *narrow band* ancora non passati a *accepted*.

4.2 Metodi semilagrangiani con *narrow band* dinamica

In questo paragrafo proponiamo alcune idee per un nuovo metodo, che eredita l'idea di base della *narrow band*, ma cambia radicalmente il modo con cui essa segue l'evolversi del fronte. L'idea fondamentale può essere così descritta:

1. Inizializzare la *narrow band*.
2. Convertire ad *accepted* tutti i nodi della *narrow band* contemporaneamente.
3. Etichettare come *narrow band* un nuovo insieme di nodi.
4. Calcolare il valore v_h dei nodi della *narrow band* con una procedura di punto fisso seguendo lo schema semilagrangiano.
5. Tornare al punto 2.

Da questa descrizione si può immediatamente osservare un vantaggio rispetto ai metodi precedentemente descritti. Infatti, viene a mancare la ricerca del minimo valore di v tra i nodi della *narrow band*. Questo permette di risparmiare un numero di operazioni pari a $O(\ln(N^2))$, e raggiungere quindi un ordine di $O(N^2)$ su una griglia bidimensionale, che è ovviamente il miglior risultato teoricamente possibile.
Tenendo presente i risultati ottenuti nei capitoli precedenti, l'unica operazione nuova che dobbiamo definire è il punto 3, cioè dobbiamo definire il procedimento che permette di aggiornare la *narrow band*. A questo proposito, abbiamo preso spunto dalla definizione di *narrow band* data in [13], cioè

$$\mathcal{R}_n^h = \left\{ x \in \mathbb{R}^n \backslash \bigcup_{j=0}^{n-1} \mathcal{R}_j \ : \ \exists\ a \in A \ \text{ tale che } \ x - hc(x)a \in \mathcal{R}_{n-1}^h \right\} \qquad (4.3)$$
$$n = 1, 2, \ldots$$

e $\mathcal{R}_0^h = \Omega$. Essa era stata usata al solo scopo di dimostrare la convergenza del metodo semilagrangiano "classico", senza essere realmente messa in pratica nell'algoritmo. Purtroppo, calcolare $\mathcal{R}_n^h$ a partire da $\mathcal{R}_{n-1}^h$ seguendo alla lettera la definizione (4.3) risulta essere molto costoso computazionalmente. D'altro canto, aggiornare la *narrow band* includendo semplicemente tutti i nodi *adiacenti* a tutti i nodi che vengono convertiti ad *accepted*, non tiene conto del campo di velocità $c(x)$ e quindi ovviamente la *narrow band* non risulta essere una approssimazione delle curve di livello della soluzione. Così

facendo, infatti, le *narrow band* non sono altro che cerchi concentrici ed esse non seguono il profilo del fronte durante la sua evoluzione.

L'idea centrale è quella di permettere alla *narrow band* di avere uno "spessore" maggiore là dove il campo di velocità è più alto, ottenendo una *narrow band dinamica* che tenga quindi conto del modulo (ed eventualmente anche della direzione) del campo. In questo modo è forse possibile considerare il caso più generale del problema di *tempo minimo*, considerando cioè un campo $b(x, a)$ generico.
I problemi che nascono immediatamente sono legati a quelli già incontrati nei metodi studiati in precedenza. In particolare, ci riferiamo all'interpolazione del valore di un punto che non coincide con un nodo, per il quale si deve poter escludere che entri in gioco un valore fittizio (ad esempio $v = 2$), e all' "esattezza" del valore dei nodi che vengono convertiti ad *accepted*. Ad esempio, è possibile che, dopo aver accettato come definitivi tutti i nodi della *narrow band*, si apra un percorso a partire da questi nodi che raggiunge nodi già *accepted* in un tempo minore di quello già calcolato.

La strada da seguire, quindi, sembra essere quella di non convertire ad *accepted* tutti i nodi della *narrow band* $\mathcal{R}_n^h$, ma tramandare a $\mathcal{R}_{n+1}^h$ i nodi di $\mathcal{R}_n^h$ che ancora non sono "sicuri", secondo un criterio da definire che dovrà tenere conto del campo di velocità.

Nel prossimo capitolo, dedicato in parte al confronto dei metodi studiati, abbiamo realizzato il metodo su descritto nel semplice caso in cui $c(x) \equiv 1$ per ogni x. In questo semplice caso le *narrow band*

$$\mathcal{R}_n^h, \quad n = 1, 2, \ldots$$

non sono altro che corone circolari di "spessore" Δx. È possibile quindi definire $\mathcal{R}_n^h$ semplicemente come l'insieme dei nodi *vicini* ai nodi di $\mathcal{R}_{n-1}^h$. Inoltre, abbiamo eseguito una sola iterazione di punto fisso per ogni *narrow band*. In questo modo si raggiunge il costo computazionale più basso teoricamente possibile. Purtroppo al momento non c'è garanzia che questi risultati possano essere mantenuti in casi più generali.

Capitolo 5

Test numerici ed applicazioni

I programmi sono stati realizzati con MATLAB 5.3 presso il centro di calcolo del dipartimento di matematica dell'università "La Sapienza" di Roma ed eseguiti su un PC dotato di processore Pentium® I, 266 MHz e 96 MB di RAM.
Abbiamo confrontato i diversi metodi numerici testandoli su vari esempi. Nei casi in cui è nota la soluzione esatta, abbiamo calcolato l'errore in L^∞ e il tempo di CPU impiegato per il calcolo.
Il dominio nel quale approssimiamo la funzione T è $Q = [-2, 2]^2$, salvo diversa indicazione.
Così come per il FMM-SL, anche nel metodo SL abbiamo scelto un passo h variabile in modo tale che $h_i c_i = \Delta x$ per ogni i. Inoltre, la ricerca del minimo è stata sempre effettuata con il metodo "veloce" descritto nel capitolo 3.
Negli esempi proposti, il fronte iniziale Γ_0 è dato tramite equazioni parametriche o è ridotto ad un singolo punto. Osserviamo che quest'ultimo caso, pur non verificando le ipotesi previste dalla teoria, può essere trattato senza problemi da tutti gli algoritmi studiati dal momento che essi non richiedono l'esistenza di nodi interni al fronte iniziale. Il calcolo dei nodi che costituiscono il fronte iniziale è incluso nel tempo di CPU indicato.
In questo paragrafo tutte le figure sono state ottenute considerando una griglia di 50×50 nodi, salvo diversa indicazione. Per il calcolo dell'errore e del tempo di CPU impiegato per l'esecuzione abbiamo considerato griglie di 25×25 nodi, 50×50 nodi e 100×100 nodi, corrispondenti rispettivamente a $\Delta x = 0.1667$, $\Delta x = 0.0816$ e $\Delta x = 0.0404$.
Osserviamo infine che nei quattro test che seguono la condizione CFL data dalla (2.10) è sempre verificata.

Test 1

$$\Gamma_0 = (0,0)\,, \quad c(x,y) \equiv 1$$

Soluzione esatta:

$$T(x,y) = \sqrt{(x^2+y^2)}$$

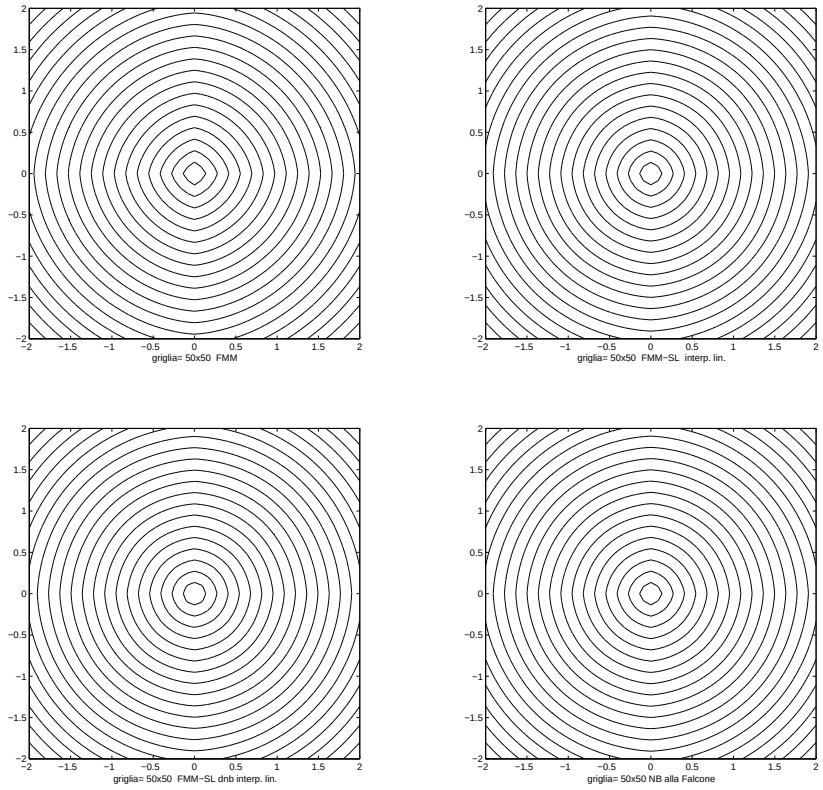


Figura 5.1: dall'alto a sinistra: FMM, FMM-SL, FMM-SL doppia *narrow band* , FMM-SL *narrow band* dinamica

metodo	Δx	errore L^∞	tempo CPU (sec)
SL (25 iteraz.)	0.1667	0.0513	23.68
FMM	0.1667	0.1455	3.02
FMM-SL	0.1667	0.0513	6.92
FMM-SL doppia nb	0.1667	0.0378	3.85
FMM-SL nb dinam	0.1667	0.0513	1.54

metodo	Δx	errore L^∞	tempo CPU (sec)
SL (50 iteraz.)	0.0816	0.0329	196.64
FMM	0.0816	0.0875	16.31
FMM-SL	0.0816	0.0329	34.28
FMM-SL doppia nb	0.0816	0.0259	20.82
FMM-SL nb dinam	0.0816	0.0329	6.26

metodo	Δx	errore L^∞	tempo CPU (sec)
SL (100 iteraz.)	0.0404	0.0204	1552.35
FMM	0.0404	0.0526	94.09*
FMM-SL	0.0404	0.0204	175.82
FMM-SL doppia nb	0.0404	0.0168	121.66
FMM-SL nb dinam	0.0404	0.0204	24.44

*5.53 sec su un PC, Processore Pentium® IV, 1.60 GHz, 256 MB RAM

Test 2

Γ_0 = due circonferenze di raggio $R = 1/2$ e centro rispettivamente

$$C_1 = (-1, 0) \text{ e } C_2 = (1, 0).$$

$$c(x, y) \equiv 1$$

Soluzione esatta:

$$T(x, y) = \min\left\{\sqrt{(x-1)^2 + y^2} - R, \sqrt{(x+1)^2 + y^2} - R\right\}$$

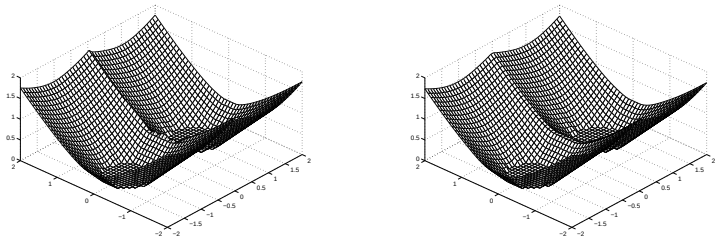


Figura 5.2: FMM (a sinistra) e FMM-SL (a destra): $T(x, y)$

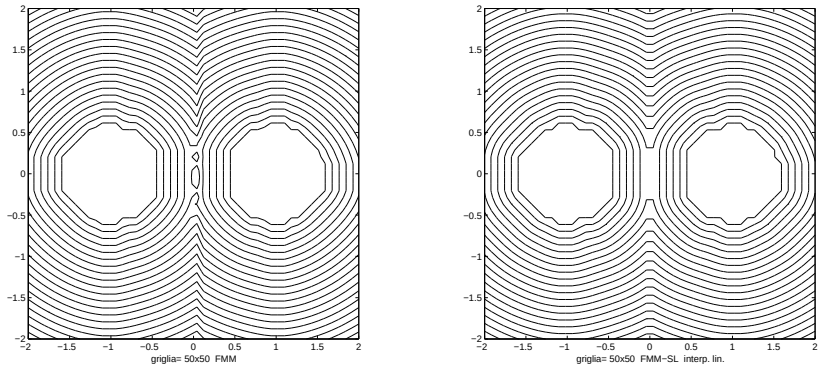


Figura 5.3: FMM (a sinistra) e FMM-SL (a destra)

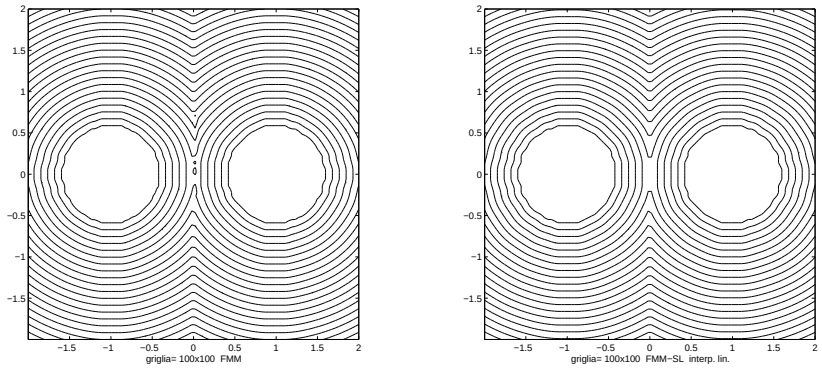


Figura 5.4: FMM (a sinistra) e FMM-SL (a destra): 100×100 nodi

metodo	Δx	errore L^∞	tempo CPU (sec)
FMM	0.1667	0.1009	3.24
FMM-SL	0.1667	0.1009	6.21
FMM-SL doppia nb	0.1667	0.1009	3.79

metodo	Δx	errore L^∞	tempo CPU (sec)
FMM	0.0816	0.1086	16.65
FMM-SL	0.0816	0.0495	31.8
FMM-SL doppia nb	0.0816	0.0495	20.32

metodo	Δx	errore L^∞	tempo CPU (sec)
FMM	0.0404	0.0439	110.07
FMM-SL	0.0404	0.0273	173.18
FMM-SL doppia nb	0.0404	0.0273	127.27

Test 3

$$\Gamma_0 = \text{quadrato di lato 1 centrato in } (0,0).$$

$$c(x,y) \equiv 1$$

Soluzione esatta:

$$T(x,y) = \begin{cases} |x| - 0.5 & |x| > 0.5\,, |y| < 0.5 \\ |y| - 0.5 & |y| > 0.5\,, |x| < 0.5 \\ \sqrt{(|x| - 0.5)^2 + (|y| - 0.5)^2} & |x| > 0.5\,, |y| > 0.5 \end{cases}$$

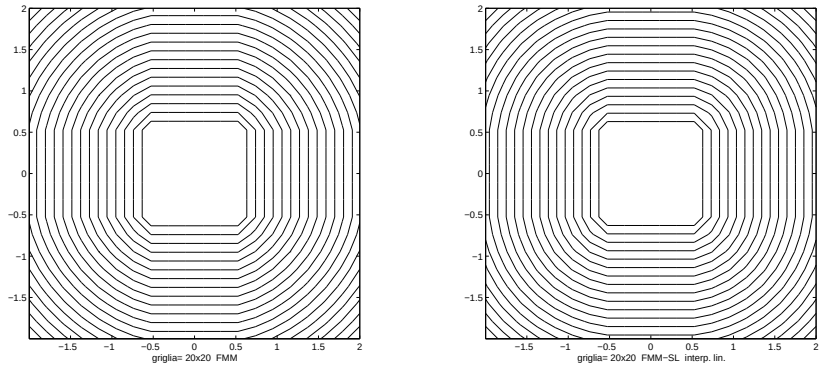


Figura 5.5: FMM (a sinistra) e FMM-SL (a destra): 20×20 nodi

metodo	Δx	errore L^∞	tempo CPU (sec)
FMM	0.1667	0.1319	2.97
FMM-SL	0.1667	0.0513	6.48
FMM-SL doppia nb	0.1667	0.0370	3.68

metodo	Δx	errore L^∞	tempo CPU (sec)
FMM	0.0816	0.0809	15.6
FMM-SL	0.0816	0.0433	30.16
FMM-SL doppia nb	0.0816	0.0260	18.46

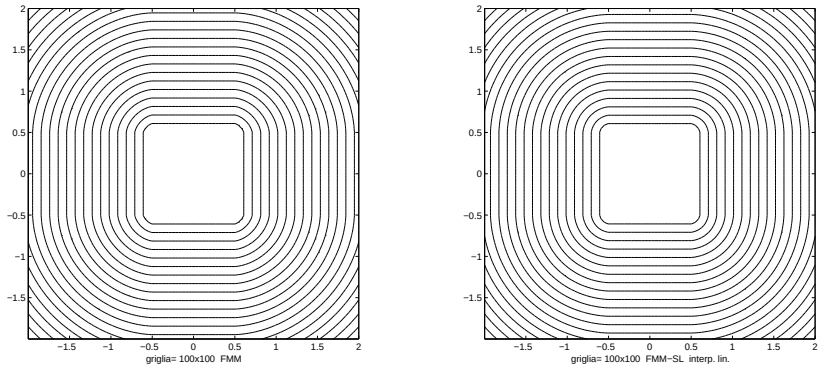


Figura 5.6: FMM (a sinistra) e FMM-SL (a destra): 100×100 nodi

metodo	Δx	errore L^∞	tempo CPU (sec)
FMM	0.0404	0.0420	91.94
FMM-SL	0.0404	0.0134	158.51
FMM-SL doppia nb	0.0404	0.0098	109.9

Commenti

Come era prevedibile dalle precedenti osservazioni teoriche, l'errore è in ogni caso dell'ordine di Δx e diminuisce in maniera regolare infittendo i nodi della griglia. Si osserva però che l'errore commesso dal FMM è sempre maggiore di quello commesso da tutti gli altri metodi basati su una approssimazione semilagrangiana. Anche i tempi di CPU rispecchiano i risultati attesi. Infatti, escludendo il metodo SL che ha un costo decisamente maggiore degli altri, tutti i metodi basati sulla tecnica della *narrow band* hanno un costo confrontabile.
Un altro importante risultato è visibile nel Test 1, dal quale si può desumere che il metodo SL e il metodo FMM-SL forniscono in output esattamente la stessa matrice di valori, risultato confermato anche da una verifica diretta.
L'unico risultato che si discosta dai risultati teorici è quello riguardante il metodo FMM-SL con doppia *narrow band*. Anch'esso avrebbe dovuto calcolare la stessa matrice di valori del metodo SL ma i risultati sono invece addirittura migliori, con un costo computazionale molto simile a quello del FMM. Questo metodo deve però essere testato su esempi in cui la velocità non è costante in modo tale da utilizzare effettivamente la seconda *narrow band*. Si veda a questo proposito l'esempio 3 nelle pagine seguenti.

In conclusione, tenendo conto dell'errore, del costo computazionale e della versatilità, il FMM-SL con doppia *narrow band* risulta essere il metodo migliore tra tutti quelli studiati.

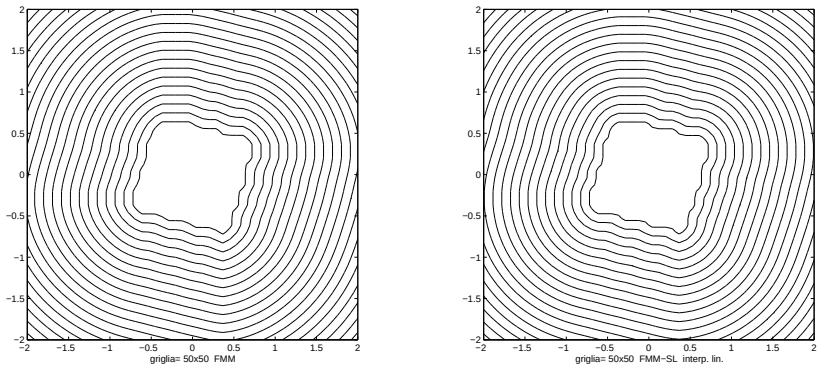
Per quanto riguarda i risultati grafici, è possibile notare l' "effetto griglia" di cui sono affetti i metodi alle differenze finite (in questo caso solamente il metodo FMM) visibile nelle quattro direzioni N,S,E,W a partire dal fronte iniziale. Nel Test 2, invece, è ben visibile la maggiore precisione dei metodi semilagrangiani, che risalta nel caso di fusioni di due o più fronti o nel caso di brusche variazioni della velocità di propagazione.

Altri esempi

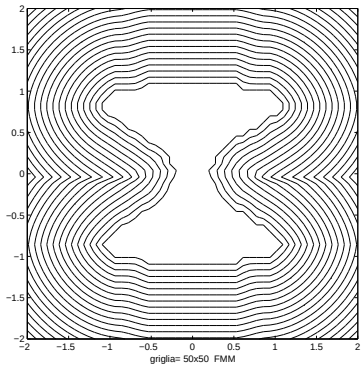
Presentiamo qui di seguito altri tre esempi di cui non si conosce l'espressione esplicita della soluzione esatta. Vengono confrontati i risultati grafici del FMM (a sinistra) e del FMM-SL (a destra).

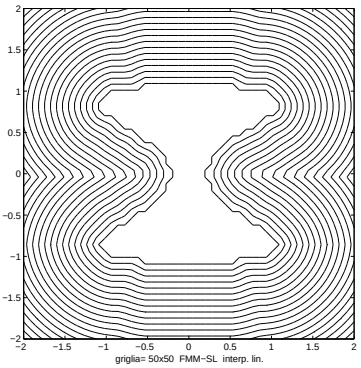
1. Γ_0 = quadrato di lato 1 centrato in $(0,0)$ ruotato di $15°$ in senso antiorario intorno all'origine.

$$c(x,y) \equiv 1$$



2.
$$\Gamma_0 = (\sin(3\sin(\theta)), \cos(\theta)), \quad \theta \in [0, 2\pi)$$
$$c(x,y) \equiv 1$$

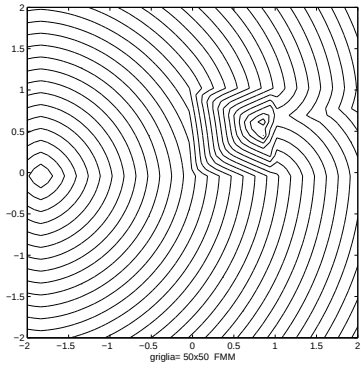


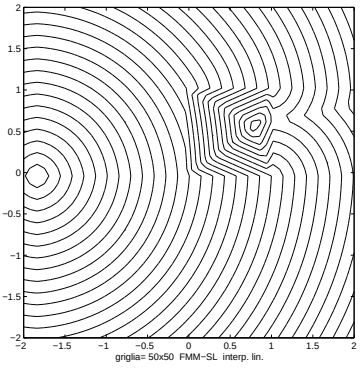


3.

$$\Gamma_0 = (-1.8, 0)$$

$$\begin{cases} c(x,y) = 0.4 & 0 < x < 1\,, \quad 0 < y < 1 \\ c(x,y) = 1 & \text{altrove} \end{cases}$$





In questo esempio la condizione CFL data dalla (2.10) non è verificata. In effetti, durante l'esecuzione, l'algoritmo si trova a dover calcolare la radice quadrata di un numero negativo. È comunque possibile portare a termine il calcolo sostituendo il radicando con il valore 0 e considerando l'errore commesso come parte dell'errore complessivo dello schema numerico.
Come si vede dalle due figure mostrate, in questo caso la differenza tra i due metodi è abbastanza marcata. Infatti, nel caso di brusche variazioni del campo di velocità, il FMM-SL è in grado di dare risultati molto più accurati, potendo "seguire" anche le direzioni oblique e non necessitando di particolari condizioni che leghino Δx e c.
Questo esempio è stato testato anche con il FMM-SL con doppia *narrow band*. Dal punto di vista grafico, esso ha dato risultati identici al FMM-SL.

5.1 Problemi di tempo minimo

Lo scopo dei problemi di *tempo minimo* è quello di ricostruire le traiettorie ottime che congiungono un punto qualsiasi $x \in Q \backslash \mathcal{T}$ con il *target* $\mathcal{T}$. Di conseguenza, dopo aver ricostruito la funzione valore $T(x)$, è necessario un ulteriore lavoro per risolvere la seguente equazione differenziale *ordinaria*

$$\begin{cases} \dot{y}(t) = c(y(t))a^*(t) \\ y(0) = x \end{cases} \tag{5.1}$$

dove $a^*(t)$ è il *controllo ottimo* (open-loop). È importante osservare che il punto iniziale x può essere scelto *dopo* la ricostruzione della funzione T. Infatti, dalla conoscenza dei valori di T su una griglia, è possibile ricostruire tutte le traiettorie ottime che raggiungono il *target* da un qualsiasi punto del suo dominio di definizione.
Introduciamo un passo di discretizzazione Δt e indichiamo con

$$y_k = y(k\Delta t)\,, \qquad a_k^* = a^*(k\Delta t).$$

Possiamo quindi facilmente ottenere dalla (5.1) il seguente schema numerico esplicito del primo ordine

$$\begin{cases} y_{k+1} = y_k - \Delta t c(y_k) a_k^* \\ y_0 = x \end{cases} \tag{5.2}$$

con

$$a_k^* = \arg\min_{a \in B(0,1)} \left\{ \beta v_h(y_k - \Delta t c(y_k) a) + 1 - \beta \right\} = \arg\min_{a \in B(0,1)} \left\{ T(y_k - \Delta t c(y_k) a) \right\} \tag{5.3}$$

La (5.3) può essere risolta con lo stesso procedimento con cui lo si è fatto durante la costruzione della funzione T, ad esempio tramite la discretizzazione di $B(0,1)$. Non è però più possibile utilizzare il metodo "veloce" descritto per il FMM-SL, in quanto esso richiede che i punti y_k coincidano con i nodi della griglia, condizione che non può più essere garantita (vedi Figura 5.7). Osserviamo che il passo di discretizzazione Δt è completamente slegato dal passo h e dal passo Δx scelti precedentemente e che lo schema numerico (5.2) può essere sostituito con un qualsiasi altro schema di ordine superiore.

Un altro modo per ricostruire le traiettorie ottime è attraverso un'approssimazione del gradiente, seguendo l'idea del FMM. Approssimando (con un metodo di qualsiasi ordine) il vettore $\nabla T(y_k)$ si può definire

$$\begin{cases} y_{k+1} = y_k - \Delta t \ \nabla T(y_k) \\ y_0 = x \end{cases} \tag{5.4}$$

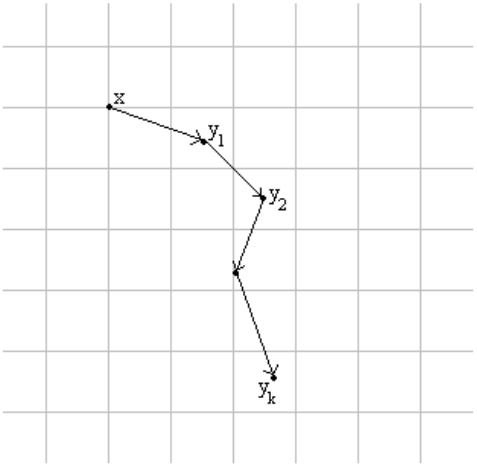
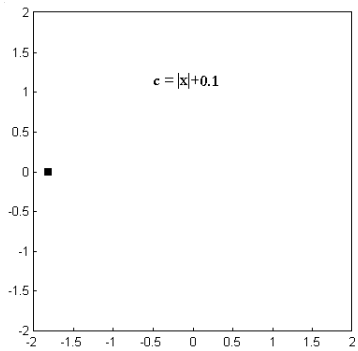
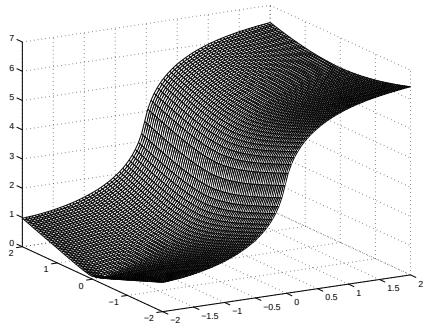


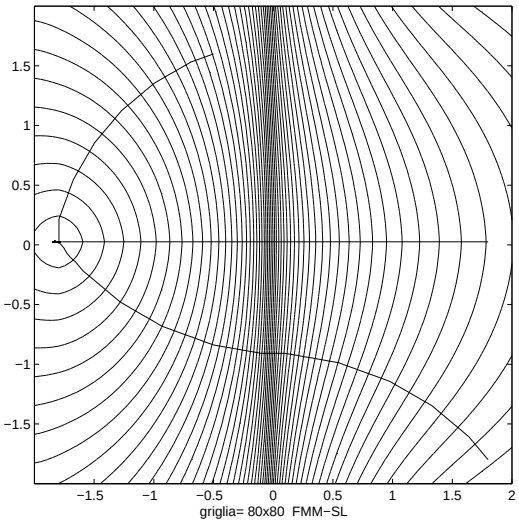
Figura 5.7: ricerca delle traiettorie ottime

Quest'ultima espressione è la traduzione analitica del fatto che le traiettorie ottime sono sempre ortogonali alle curve di livello della funzione T.

Presentiamo qui di seguito alcuni esempi, realizzati con il metodo FMM-SL su una griglia di 80×80 nodi. Per il calcolo di a_k^* si è usata una discretizzazione di $B(0,1)$ ottenuta tramite 36 controlli posti sulla frontiera (uno ogni $10°$).
Per ogni esempio mostriamo:
- la raffigurazione del dominio Q con il *target* $\mathcal{T}$ e la funzione $c(x)$.
- la funzione valore T.
- le curve di livello di T con alcune traiettorie ottime.



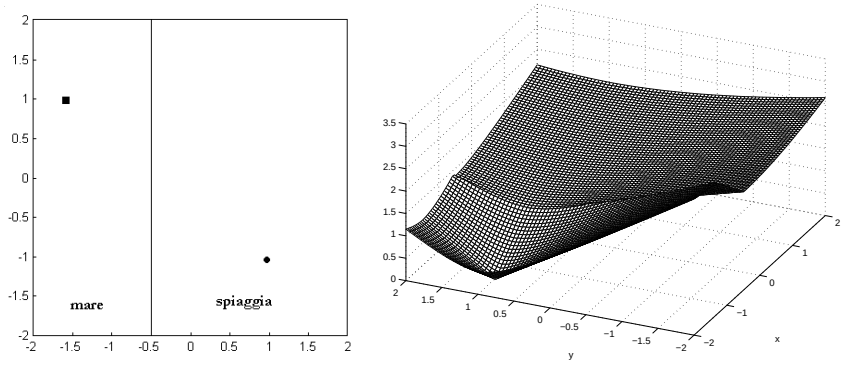


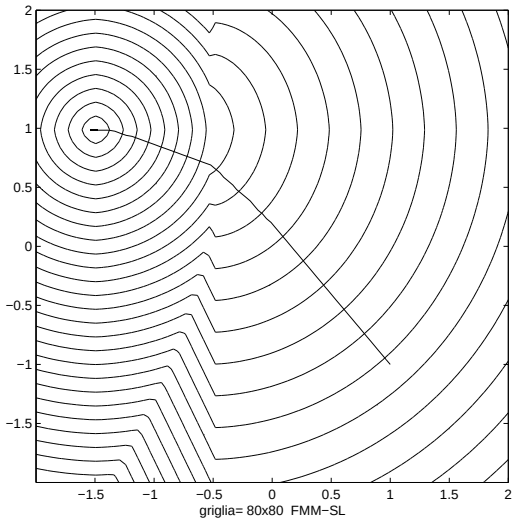


In questo esempio si ha

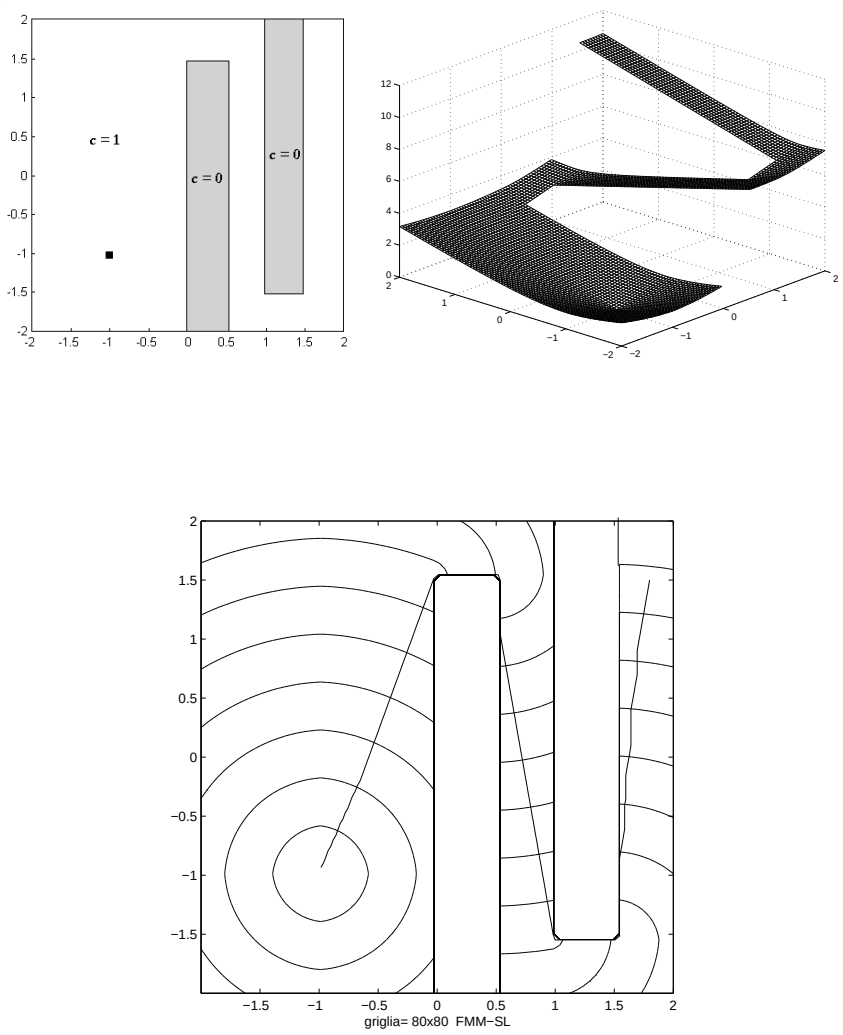
$$c(x, y) = |x| + 0.1$$

Abbiamo quindi un campo di velocità regolare (escludendo la retta $x = 0$) che produce una significativa variazione nei tempi di primo arrivo all'interno del dominio.
Si osservi come la traiettoria di punto iniziale $(1.8, -1.8)$ cerchi di minimizzare la permanenza nella striscia $\{-0.5 < x < 0.5\}$ nella quale la velocità di percorrenza è più bassa.

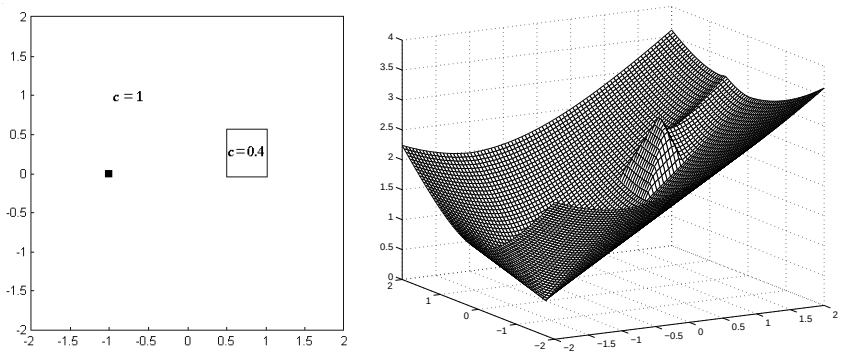


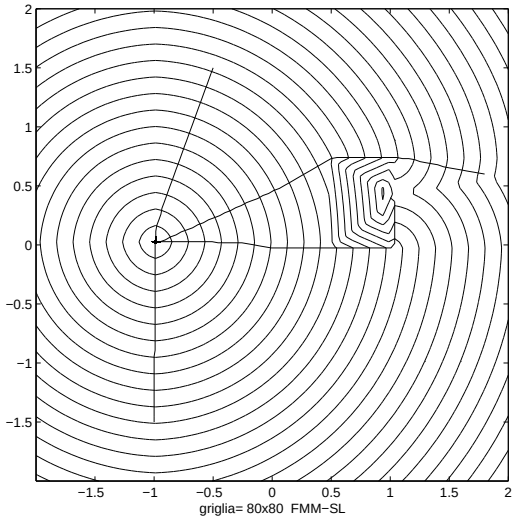


L'esempio mostra la traiettoria migliore che deve fare un bagnino per salvare un bagnante che sta affogando. Il fatto che essa non sia una linea retta è dovuto al fatto che la velocità alla quale si corre sulla spiaggia è maggiore della velocità alla quale si nuota nell'acqua.
L'angolo formato dalla traiettoria ottima nel punto corrispondente a $x = -0.5$ segue la nota *legge della rifrazione* (o di Cartesio-Snell) formulata nell'ambito dell'ottica geometrica.

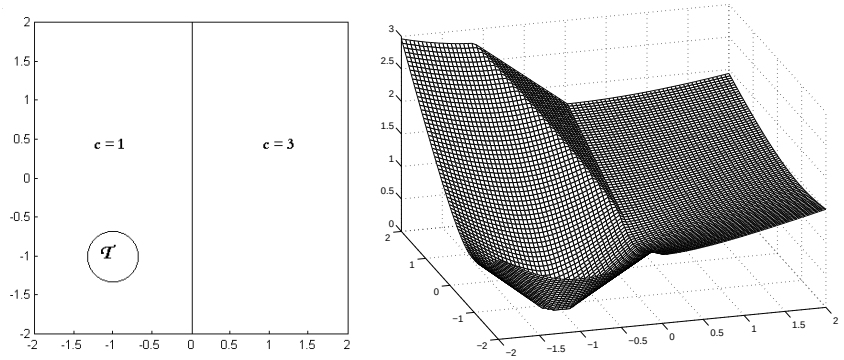


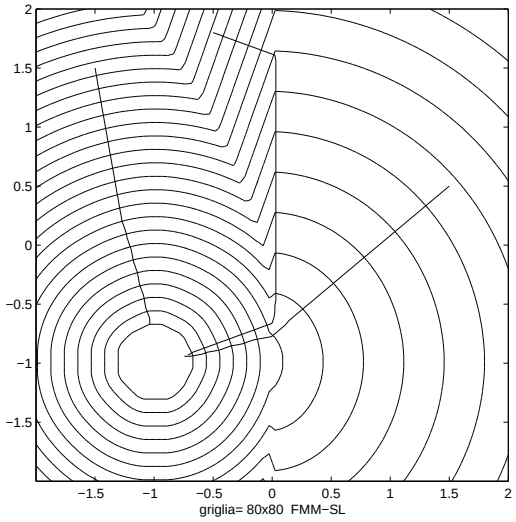
In questo esempio è possibile vedere come la trasformata di Kruzkov permette di trattare la presenza di ostacoli nel dominio Q. Inoltre, si nota il ruolo giocato dalla discretizzazione di $B(0,1)$ nell'approssimazione del *controllo ottimo*. Se la direzione della traiettoria ottima non coincide esattamente con una delle direzioni considerate, la traiettoria "numerica" appare irregolare. Un esempio è ben visibile nella striscia $\{1.5 < x < 2\}$ nella quale la traiettoria ottima dovrebbe essere perfettamente rettilinea fino al raggiungimento dell'ostacolo ed invece segue un andamento a *zig-zag* saltando tra le due direzioni che approssimano meglio quella esatta.



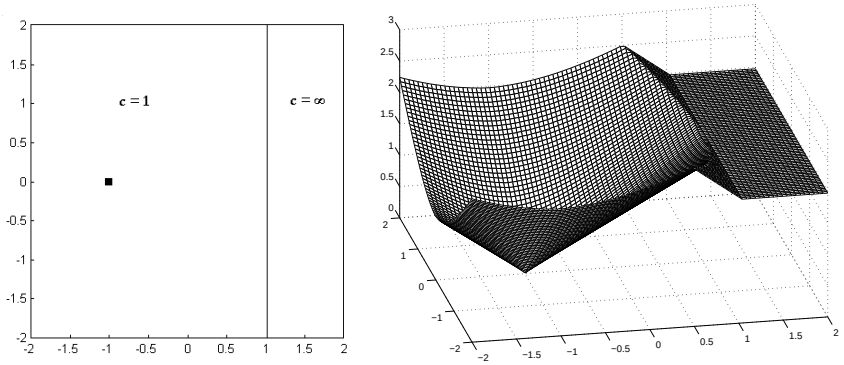


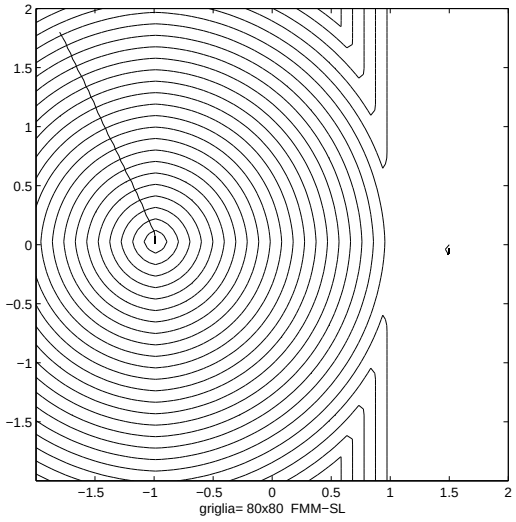
In questo esempio è interessante osservare il *picco* presente nella funzione valore in corrispondenza del quadrato nel quale la velocità è più bassa. Inoltre è possibile osservare la "scia" che il *picco* porta dietro di sé. Essa è dovuta al fatto che le traiettorie che partono da questa regione devono aggirare la zona nella quale $c(x, y) = 0.4$ con conseguente innalzamento del tempo di arrivo al *target*.





È interessante commentare la *traiettoria ottima* che congiunge il punto iniziale $(-0.5, 1.8)$ con il *target*. Essa devia verso la regione dove la velocità di spostamento è maggiore, per poi ripiegare verso il suo obiettivo finale. Un esempio dello stesso tipo è mostrato in [24].





In questo esempio è mostrato come il FMM-SL possa trattare il caso in cui la velocità di propagazione è infinita. Come si può osservare, nel momento in cui il fronte raggiunge il punto $(1, 0)$, esso si propaga istantaneamente in tutta la regione $\{x \geq 1\}$. L'algoritmo non è però in grado di ricostruire le traiettorie ottime che congiungono i punti di questa regione con il *target* poiché la soluzione $T(x, y)$ è costante per $x \geq 1$. Di conseguenza, se (x_i, y_j) è tale che $x_i > 1$, si avrà

$$T(x_{i\pm1}, y_j) = T(x_i, y_{j\pm1}) = T(x_{i\pm1}, y_{j\pm1})$$

Non è quindi possibile distinguere in alcun modo una direzione particolare. Si osservi il tentativo fatto per il punto $(1.5, 0)$.

5.1.1 Problemi di tempo minimo su superfici

Sia data una superficie

$$z : Q \subset \mathbb{R}^2 \to \mathbb{R}$$

sulla quale ci si può spostare lungo ogni direzione con velocità pari a 1. Supponiamo inoltre che sulla superficie sia definito un *target*, in modo analogo al problema di tempo minimo. L'obiettivo è quello di calcolare le *traiettorie ottime* che raggiungono il *target* da un qualsiasi punto della superficie, rimanendo sempre su di essa.
L'idea di base è quella di proiettare la superficie sul suo insieme di definizione Q, e ricondursi ad un "classico" problema di tempo minimo nel piano, analogo a quello descritto precedentemente. Questo risultato viene ottenuto definendo in modo opportuno una velocità $c : Q \to \mathbb{R}$, che dipenderà ovviamente da z. Una volta approssimata la *traiettoria ottima* nel piano, la si proietta sulla superficie per ottenere la soluzione del problema originario.

Ragioniamo su un semplice caso unidimensionale nel quale la superficie è sostituita dal grafico di una funzione lineare $z : \mathbb{R} \to \mathbb{R}$. La generalizzazione al caso bidimensionale seguirà in modo semplice.
Consideriamo i punti O, P, O' e P' come mostrato in Figura 5.8 Il tempo impiegato per andare da O a P spostandosi a velocità 1 lungo il grafico di z è uguale alla lunghezza del segmento OP, cioè

$$\overline{OP} = \sqrt{\Delta x^2 + (z'(x)\Delta x)^2}$$

La velocità con la quale si deve percorrere il segmento $O'P'$ affinché il tempo impiegato sia uguale a quello impiegato per percorrere il segmento OP è

$$c = \frac{spazio}{tempo} = \frac{\Delta x}{\sqrt{\Delta x^2 + (z'(x)\Delta x)^2}} = \frac{1}{\sqrt{1 + (z'(x))^2}}$$

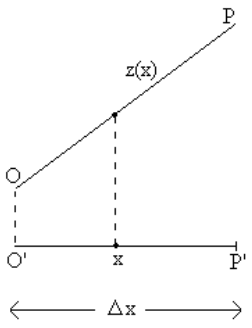


Figura 5.8: Problema di tempo minimo su superfici

In pratica, c è la velocità alla quale si sposta la proiezione sul segmento $O'P'$ di un punto materiale che percorre il segmento OP.
Per quanto riguarda il caso bidimensionale, è sufficiente sostituire $z'(x)$ con la derivata direzionale di z lungo la direzione a, ottenendo

$$c(x, y, a) = \frac{1}{\sqrt{1 + (\nabla z \cdot a)^2}} \tag{5.5}$$

La dipendenza da a nel modulo della velocità è fondamentale e cambia notevolmente le proprietà della soluzione rispetto ai casi precedentemente studiati, rientrando nei casi di propagazione *anisotropa* dei fronti. Infatti, la propagazione del fronte avviene ancora in direzione della normale esterna, ma il modulo della velocità dipende dalla normale stessa. Questo fa sì che le *traiettorie ottime* non siano più ortogonali alle curve di livello della funzione valore T. La situazione è chiarita nell'esempio in Figura 5.9. Indichiamo con

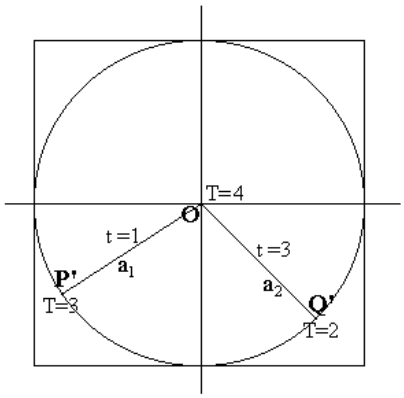


Figura 5.9: caso anisotropo

$t_{OP'}$ e $t_{OQ'}$ rispettivamente il tempo impiegato a percorrere i segmenti OP' e OQ'. Sebbene le lunghezze dei due segmenti siano uguali, i valori $t_{OP'}$ e $t_{OQ'}$ saranno in generale diversi, perché essi vengono percorsi a velocità differenti. Come è facile vedere, sebbene si abbia $T(Q') < T(P')$, la direzione a_1 è preferibile alla direzione a_2 in quanto

$$T(P') + t_{OP'} < T(Q') + t_{OQ'}.$$

La ricerca del *controllo ottimo* deve quindi essere necessariamente realizzata tramite una discretizzazione di $B(0, 1)$ (noi abbiamo usato 36 controlli), e allo stesso modo deve essere ricostruita la *traiettoria ottima* a partire dalla funzione valore. Osserviamo ancora una volta che è necessario calcolare la

velocità di propagazione del fronte per ogni controllo preso in considerazione. Si rimanda a [27] per una trattazione più completa e rigorosa della propagazione anisotropa dei fronti ed a [18] e [24] per degli esempi.

Per quanto riguarda la ricerca di geodetiche su superfici, ci sono molti campi di ricerca ancora aperti. Innanzitutto, si deve provare la validità del FMM-SL nel caso anisotropo. Come mostrato in [27], questo potrebbe presentare numerose difficoltà, dal momento che non si ha più la garanzia che l'ordine con cui i nodi vengono convertiti ad *accepted* corrisponda ad un ordine crescente per i valori di T (o di v).
In secondo luogo, i risultati ottenuti numericamente dovrebbero essere confrontati con i numerosi risultati teorici che la geometria differenziale mette a disposizione per questo tipo di problemi.

Nelle pagine seguenti mostriamo i risultati ottenuti su tre diverse superfici. Nei primi due esempi riportiamo il grafico della funzione valore T, le sue curve di livello con le traiettorie ottime calcolate e il grafico della superficie z con le traiettorie ottime disegnate su di essa. Nel terzo esempio riportiamo anche il confronto tra la traiettoria ottima calcolata numericamente e quella esatta.

Osserviamo che le traiettorie ottime *non* sono in generale ortogonali alle curve di livello della funzione T, così come previsto dalla teoria.
La griglia è scelta in tutti i casi di 100×100 nodi.

Test 1

$$z = sen(3x)sen(3y)$$

$$Q = [-1.5, 1.5] \qquad \mathcal{T} = (0, 0)$$

Punti iniziali: $(0.4, 0.4)$, $(1, 1)$, $(-1.3, -0.5)$.

Test 2

$$z = \begin{cases} 1 - (|x| + |y|) & |x| + |y| < 1 \\ 0 & \text{altrimenti} \end{cases}$$

$$Q = [-1.5, 1.5] \qquad \mathcal{T} = (0, -0.6)$$

Punto iniziale: $(0.1, 0.5)$.

Test 3

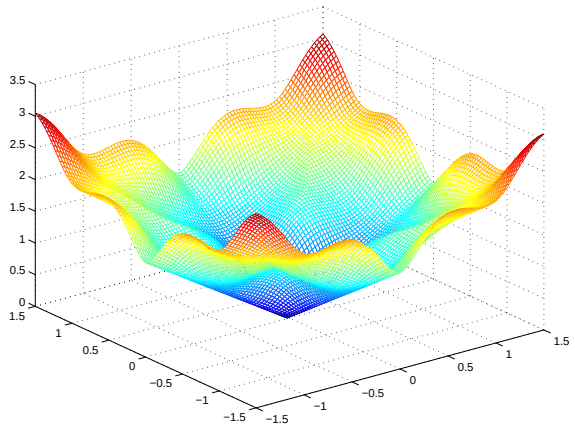
$$z = \begin{cases} \sqrt{1 - x^2 - y^2} & x^2 + y^2 < 1 \\ 0 & \text{altrimenti} \end{cases}$$

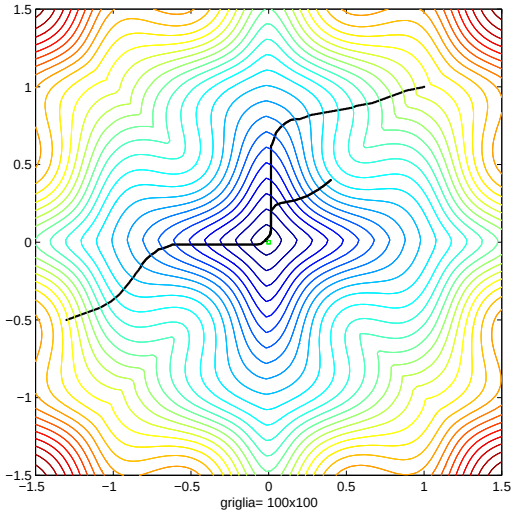
$$Q = [-1.2, 1.2] \qquad \mathcal{T} = (0, -0.8)$$

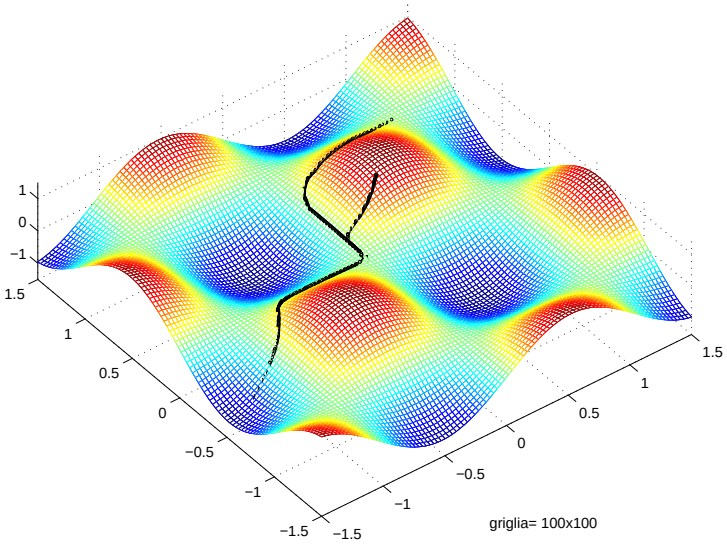
Punto iniziale: $(-0.8, 0)$.

In questo caso, nel quale la superficie z è una semisfera di raggio 1 e centro $(0, 0, 0)$, è noto che le geodetiche sono archi di cerchio di raggio 1. Di conseguenza, abbiamo sovrapposto all'immagine "calcolata", il piano passante per il punto $(0, 0, 0)$, il *target* ed il punto iniziale della traiettoria. Così facendo, l'intersezione tra il piano e la sfera risulta essere esattamente la geodetica "esatta", che può essere quindi confrontata con quella "calcolata". Come si può vedere il risultato è più che soddisfacente.

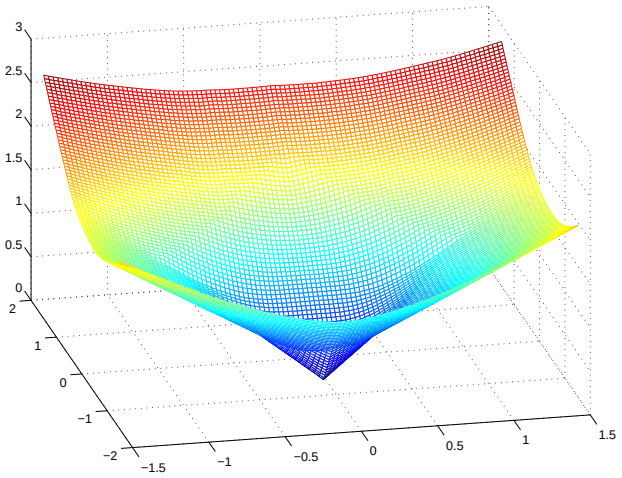
TEST 1

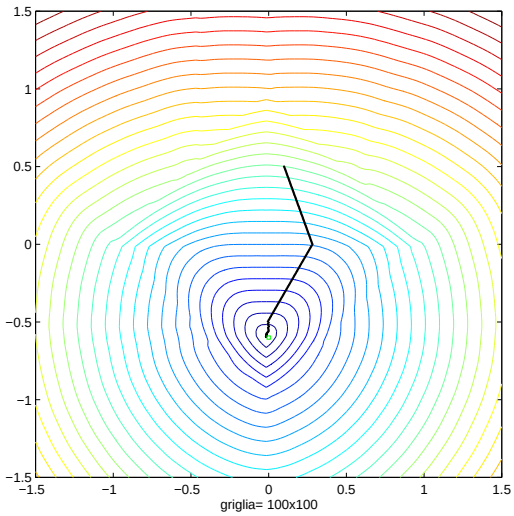

3.5
3
2.5
2
1.5
1
0.5
0
1.5
1
0.5
0
−0.5
−1
−1.5
−1.5
−1
−0.5
0
0.5
1
1.5

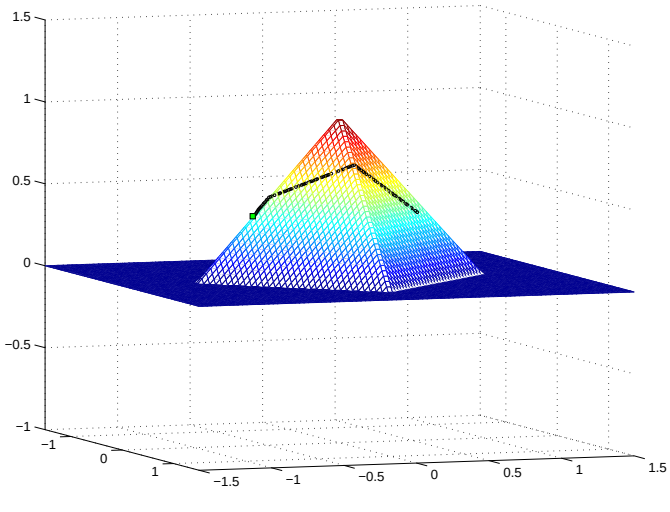

1.5
1
0.5
0
−0.5
−1
−1.5
−1.5
−1
−0.5
0
0.5
1
1.5
griglia= 100x100


1
0
−1
1.5
1
0.5
0
−0.5
−1
−1.5
−1.5
−1
−0.5
0
0.5
1
1.5
griglia= 100x100

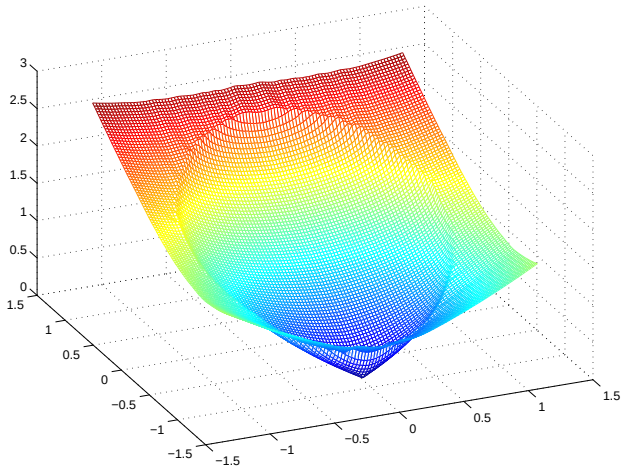
TEST 2

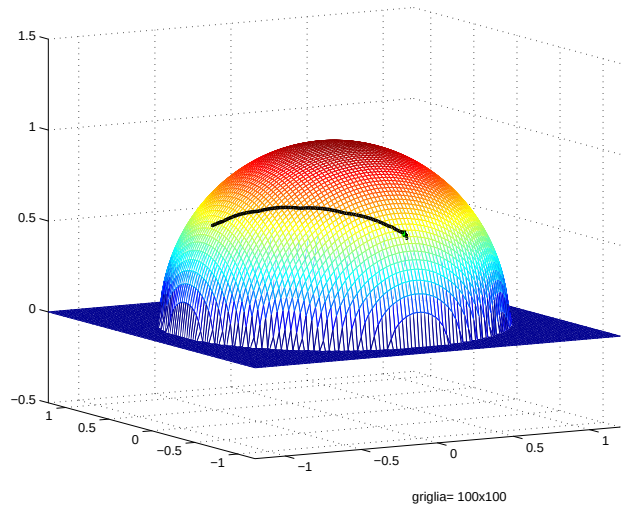

3
2.5
2
1.5
1
0.5
0
2
1
0
−1
−2
−1.5
−1
−0.5
0
0.5
1
1.5


1.5
1
0.5
0
−0.5
−1
−1.5
−1.5
−1
−0.5
0
0.5
1
1.5
griglia= 100x100

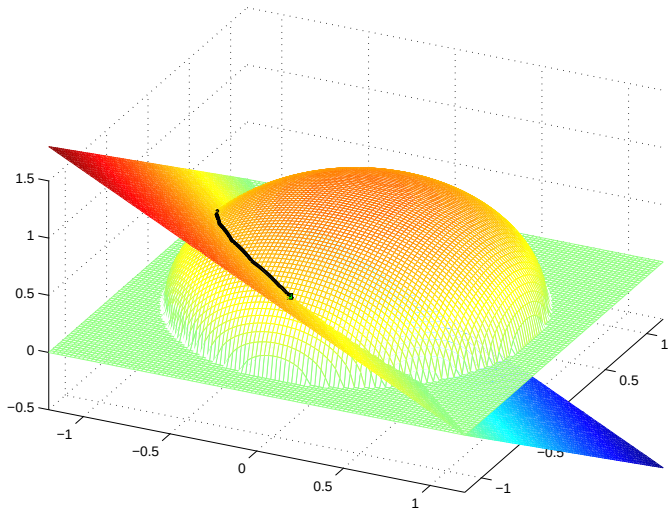

1.5
1
0.5
0
−0.5
−1
−1
0
1
−1.5
−1
−0.5
0
0.5
1
1.5
griglia= 100x100

TEST 3





griglia= 100x100



griglia= 100x100

5.2 Shape from shading

Sia data la superficie

$$T : Q \subset \mathbb{R}^2 \to \mathbb{R}.$$

Supponiamo che essa sia illuminata da un'unica sorgente luminosa proveniente dalla direzione (α, β, γ) e posta a distanza infinita da essa, dimodoché i raggi luminosi possano essere considerati paralleli. Supponiamo inoltre che la superficie sia lambertiana, cioè che abbia la capacità di riflettere la luce in maniera uniforme.
Lo scopo dello *shape from shading* è quello di ricostruire la superficie T dall'analisi delle zone di luce ed ombra presenti su di essa.

È noto che il versore normale alla superficie T è dato da

$$n = \frac{(-T_x, -T_y, 1)}{(1 + |\nabla T|^2)^{1/2}}$$

Definiamo la funzione I nel seguente modo:

$$I : Q \to [0, 1], \qquad I(x, y) = (\alpha, \beta, \gamma) \cdot n.$$

Essa associa ad ogni punto $(x, y) \in Q$ la luminosità della superficie T nel punto $(x, y, T(x, y))$.
Noi ci occuperemo solamente del caso di luce verticale, in cui

$$(\alpha, \beta, \gamma) = (0, 0, 1).$$

Sostituendo, otteniamo

$$I(x, y) = \frac{1}{(1 + |\nabla T|^2)^{1/2}}. \tag{5.6}$$

Riordinando i termini, si ha

$$|\nabla T| = \sqrt{\frac{1}{I^2} - 1}.$$

Infine, definendo

$$c(x, y) = \left(\sqrt{\frac{1}{I^2} - 1} \right)^{-1}$$

otteniamo l'ormai ben nota equazione

$$c(x, y)|\nabla T| = 1. \tag{5.7}$$

Consideriamo ad esempio i punti della superficie in cui il versore normale è parallelo ai raggi luminosi. Appare chiaro che essi saranno i più luminosi e in quei punti il valore della funzione I sarà uguale ad 1. Al contrario, i punti in cui il versore normale è ortogonale ai raggi luminosi saranno neri e la funzione I assumerà il valore 0.
Osserviamo che i casi $I = 0$ e $I = 1$ corrispondono rispettivamente a $c = 0$ e $c = +\infty$.

È di fondamentale importanza notare che il problema - posto in questi termini - non può essere risolto in maniera univoca. Ad esempio, si può facilmente vedere che le due superfici disegnate in Figura 5.10 hanno la stessa funzione

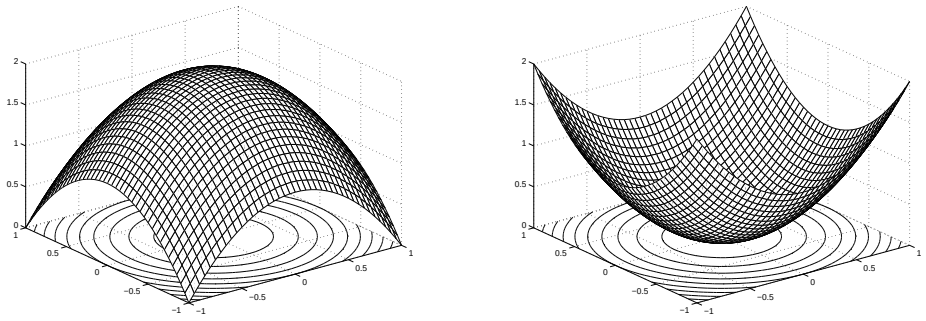


Figura 5.10: superfici corrispondenti alla stessa funzione I

di luminosità I. Il modello infatti non è in grado di distinguere tra superfici concave e superfici convesse e di conseguenza non è in grado di fornire la soluzione corretta se essa è diversa dalla soluzione di *viscosità*. Si veda a questo proposito [5] e [6] per alcuni importanti risultati teorici e numerici.

Le condizioni al bordo da affiancare alla (5.7) vanno trattate caso per caso. È possibile inoltre imporre il valore di T in qualche punto interno a Q (qualora questa informazione sia disponibile), in modo tale da selezionare la soluzione corretta.

Qui di seguito presentiamo due test realizzati con il FMM-SL su una griglia di 80×80 nodi. Nel primo abbiamo scelto $Q = [-2, 2]^2$ e

$$I(x, y) = \begin{cases} 1 & |x| < 0.4\,,\ |y| < 0.4 \\ 1/2 & \text{altrimenti} \end{cases}$$

Nel secondo test abbiamo scelto $Q = [-0.6, 0.6]^2$ e

$$I(x, y) = 1 - (x^2 + y^2)$$

In entrambi i casi si è posto

$$T(x,y) = 0 \quad (x,y) \in \partial Q$$

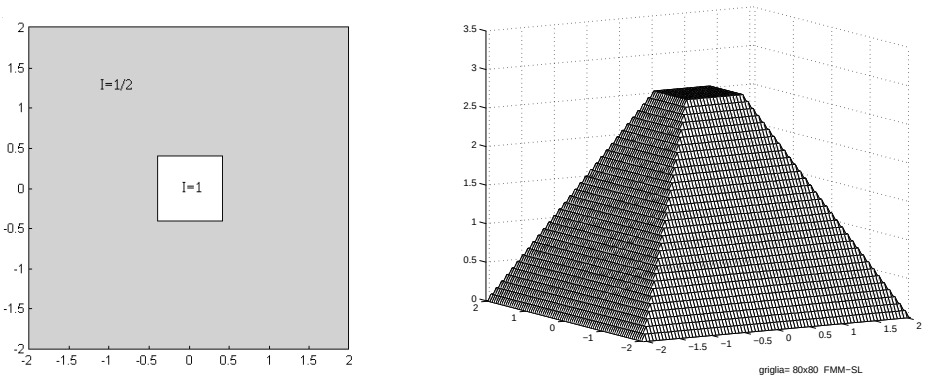


Figura 5.11: TEST 1: immagine bidimensionale e superficie ricostruita

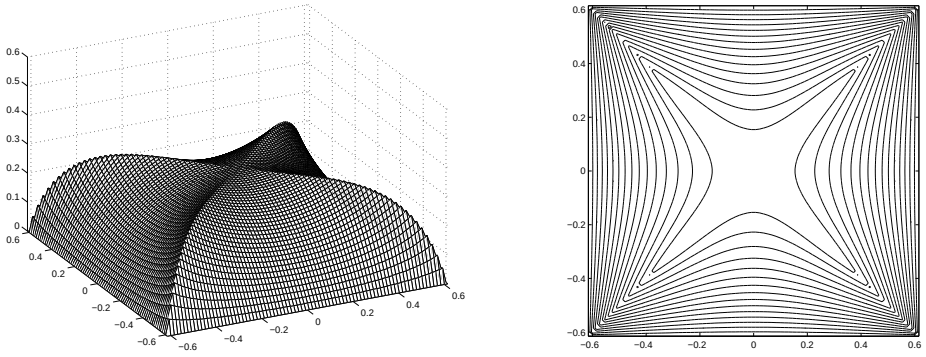


Figura 5.12: TEST 2: superficie ricostruita e sue curve di livello

Come si vede l'algoritmo approssima correttamente la soluzione di *viscosità*. Inoltre, nel TEST 1 è da notare che la superficie ricostruita è perfettamente piatta in corrispondenza del quadrato centrale nel quale $I = 1$ (cioè $c = +\infty$).

5.3 Il modello di Poincaré

Il modello di Poincaré è uno tra i più noti modelli per la geometria iperbolica, una geometria non euclidea nella quale si assume come assioma delle

parallele l'enunciato di Lobacevskij:

"*Data una retta e un punto esterno ad essa, per tale punto passano almeno due rette che non incontrano la retta data*".

Il modello di Poincaré è costruito a partire da un cerchio euclideo Σ, che noi supporremo coincidere con il cerchio di raggio 1 centrato nell'origine.
Nel modello si interpretano come *punti* i punti comunemente intesi del cerchio Σ (esclusi i punti sul bordo), e come *rette* i diametri di Σ e gli archi di cerchio interni a Σ, con estremi su $\partial\Sigma$ e ortogonali a $\partial\Sigma$ (vedi Figura 5.13).
In tale modello è possibile definire una *distanza* (*iperbolica*) ed una norma

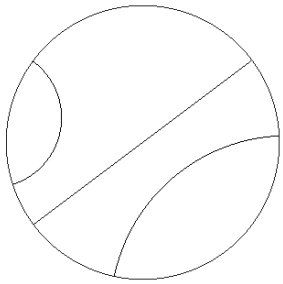


Figura 5.13: *rette* nel modello di Poincaré

$\|\cdot\|_{IP} : \Sigma \to \mathbb{R}$. Indichiamo con $\|\cdot\|_E$ la norma euclidea definita sul cerchio. È possibile dimostrare che se in ogni punto del cerchio la velocità di spostamento è

$$c(x) = 1 - \|x\|_E^2 \quad \text{per ogni} \quad x \in \Sigma\,,$$

allora le traiettorie che minimizzano il tempo di percorrenza sono le *rette* del modello di Poincaré. Infatti, in questo caso il tempo necessario a una particella per andare da P a Q lungo una curva $x(t)$ coincide precisamente con la lunghezza iperbolica di $x(t)$. Ne consegue che le traiettorie che minimizzano il tempo di percorrenza sono precisamente quelle che minimizzano la lunghezza iperbolica, cioè le *rette* nel modello.
Rimandiamo a [22] per una trattazione completa delle geodetiche nel modello di Poincaré.

Abbiamo testato il FMM-SL su questo problema ponendo

$$Q = [-1, 1]^2\,, \qquad \mathcal{T} = (-0.65, -0.65)$$

e

$$c(x,y) = \begin{cases} 1-(x^2+y^2) & (x,y) \in \Sigma \\ 0 & \text{altrimenti} \end{cases}$$

La griglia è di 130×130 nodi. Per la ricostruzione delle traiettorie ottime abbiamo usato una discretizzazione di $B(0,1)$ tramite 72 controlli. Nella pagina seguente riportiamo i risultati ottenuti.

Osserviamo che le curve di livello della funzione valore T sono *cerchi iperbolici* di centro $\mathcal{T}$, sono cioè il luogo dei *punti* equidistanti (nel senso della distanza iperbolica) da $\mathcal{T}$.

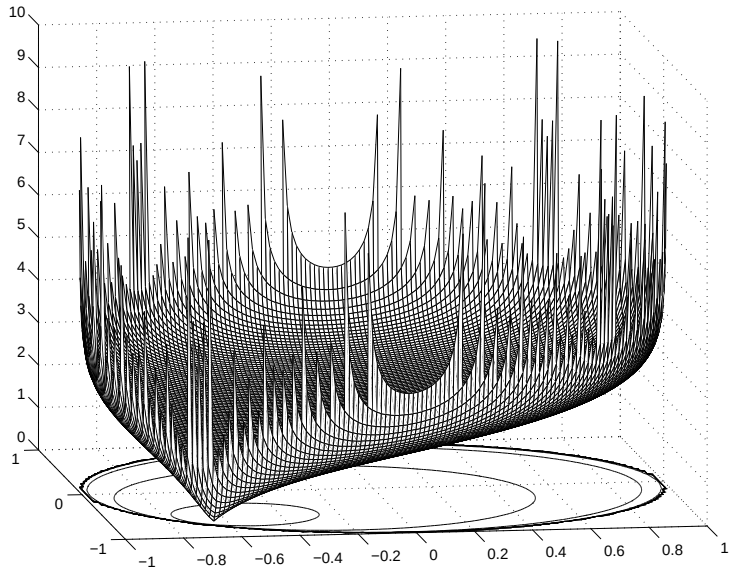


Figura 5.14: funzione valore T

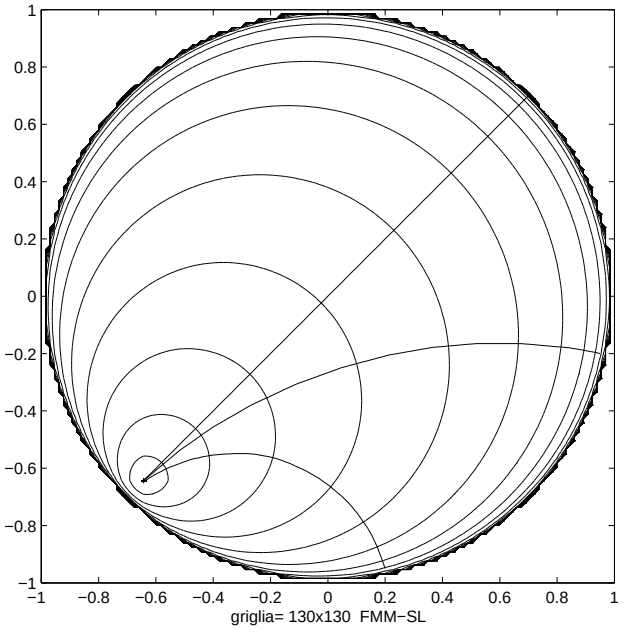


Figura 5.15: curve di livello e traiettorie ottime

Bibliografia

[1] M. Bardi, I. Capuzzo Dolcetta, *Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations*, Birkhäuser, 1997.

[2] M. Bardi, M. Falcone, *An approximation scheme for the minimum time function*, SIAM J. Control Optim., 28 (1990), pp. 950-965.

[3] G. Barles, *Solutions de viscosité des equations de Hamilton-Jacobi*, Berlin: Springer, 1994.

[4] G. Barles, P. E. Souganidis, *Convergence of approximation schemes for fully nonlinear second order equations*, Asymptotic analysis, Vol. 4 (1991), pp. 271-283.

[5] F. Camilli, A. Siconolfi, *Maximal subsolutions for a class of degenerate Hamilton-Jacobi problems*, Indiana Univ. Math. J., vol. 48 (1999), No. 3, pp. 1111-1131.

[6] F. Camilli, L. Grüne, *Numerical approximation of the maximal solutions for a class of degenerate Hamilton-Jacobi equations*, SIAM J. Numer. Anal., vol. 38 (2000), No. 5, pp. 1540-1560.

[7] E. Cotognini, *Le geometrie non euclidee: osservazioni e spunti di riflessione attraverso modelli noti e meno noti*, Tesi di laurea, 2003.

[8] M. G. Crandall, P.-L. Lions, *Viscosity solutions of Hamilton-Jacobi equations*, Tran. AMS, vol. 277 (1983), pp. 1-43.

[9] M. G. Crandall, L. C. Evans, P. L. Lions, *Some properties of Viscosity solutions of Hamilton-Jacobi equations*, Tran. AMS, vol. 282 (1984), pp. 487-502.

[10] M. G. Crandall, P.-L. Lions, *Two approximations of solutions of Hamilton-Jacobi equations*, Math. Comp. vol. 43 (1984), No. 167, pp. 1-19.

[11] L. C. Evans, *Partial differential equations*, Graduate studies in mathematics, vol. 19, AMS, 1998.

[12] L. C. Evans, P. E. Souganidis, *Differential games and representation formulas for solutions of Hamilton-Jacobi-Isaacs Equations*, Indiana Univ. Math. J. Vol. 33 (1984), No. 5, pp. 773-797.

[13] M. Falcone, *The minimum time problem and its applications to front propagation*, all'interno di *Motion by mean curvature and related topics*, Proceedings of the international conference in Trento, 1992, Walter de Gruyter, New York (1994).

[14] M. Falcone, T. Giorgi, P. Loreti, *Level sets of viscosity solution: some applications to fronts and rendez-vous problems*, SIAM J. Appl. Math., Vol. 54 (1994), No. 5, pp. 1335-1354.

[15] M. Falcone, C. Truini, *A level-set algorithm for front propagation in the presence of obstacles*, preprint.

[16] M. Falcone, *Numerical solution of dynamic programming equations*, appendice A di [1].

[17] R. Fedkiw, S. Osher, *Level set methods and dynamic implicit surfaces*, Applied Mathematical Sciences, vol. 153, Springer-Verlag, New York, 2003.

[18] R. Kimmel, J. A. Sethian, *Computing geodesic paths on manifold*, Proc. Natl. Acad. Sci. USA, Vol. 95 (1998), No. 15, pp. 8431-8435.

[19] R. Kimmel, J. A. Sethian, *Optimal algorithm for shape from shading and path planning*, Journal of mathematical imaging and vision, Vol. 14 (2001), No. 3, pp. 237-244.

[20] S. N. Kruzkov, *Generalized solutions of Hamilton-Jacobi equations of eikonal type*, I, Math USSR Sbornik, Vol. 27 (1975), pp. 406-445.

[21] P-L. Lions, *Generalized solutions of Hamilton-Jacobi equations*, Research notes in mathematics, Vol. 69, Pitman, Boston, Mass.-London, 1982.

[22] B. O'Neill, *Elementary differential geometry*, Academic Press, New York and London, 1966.

[23] E. Rouy, A. Tourin, *A viscosity solutions approach to shape-from-shading*, SIAM J. Numer. Anal., Vol. 29 (1992), No. 3, pp. 867-884.

[24] J. A. Sethian, *Level set methods, evolving interfaces in geometry, fluid mechanics, computer vision, and materials science*, Cambridge University Press, 1996.

[25] J. A. Sethian, *A fast marching level set method for monotonically advancing fronts*, Proc. Natl. Acad. Sci. USA, Vol. 93 (1996), No. 4, pp. 1591-1595.

[26] J. A. Sethian, A. Vladimirsky, *Ordered upwind methods for static Hamilton-Jacobi equations*, Proc. Natl. Acad. Sci. USA, Vol. 98 (2001), No. 20, pp. 11069-11074.

[27] J. A. Sethian, A. Vladimirsky, *Ordered upwind methods for static Hamilton-Jacobi equations: theory and algorithms*, preprint.

[28] P. Soravia, *Boundary value problems for Hamilton-Jacobi equations with discontinuous Lagrangian*, Indiana Univ. Math. J., Vol. 51 (2002), No. 2, pp. 451-477.

[29] J. N. Tsitsiklis, *Efficient algorithms for globally optimal trajectories*, IEEE Tran. Automatic. Control, Vol. 40 (1995), No. 9, pp. 1528-1538.

Ringraziamenti

Desidero ringraziare M. Sagona per aver contribuito allo sviluppo del FMM-SL. Inoltre, desidero ringraziare A. Sambusetti e E. Cotognini per le interessanti discussioni che mi hanno aiutato nella stesura del paragrafo dedicato al modello di Poincaré.