POLITECNICO DI MILANO
Scuola di Ingegneria Industriale e dell'Informazione
Laurea magistrale in Ingegneria Matematica

Volatilità locale in assenza di arbitraggio

Metodi di implementazione a confronto

Relatore:
Prof. Daniele MARAZZINA
Correlatore:
Dott. Gaetano LA BUA

Tesi di Laurea di:
Paolo TIBERTI
Matr. n.801813

Anno Accademico 2014–2015

Grazie a chiunque senta di avermi a suo modo aiutato
a raggiungere questo traguardo.

Mozzanica, settembre 2015

Sommario

Per far fronte al problema della volatilità implicita, la volatilità locale è sempre stata fin dalla sua prima formulazione nel 1994 uno tra gli strumenti più utilizzati nel mondo finanziario per il calcolo dei prezzi di derivati esotici e strutturati.

Diversi modelli e teorie sono state proposte successivamente per migliorarne l'utilizzo e aggirare alcuni dei suoi lati negativi. In questa tesi verranno esposti due dei principali modelli per la generazione della superficie di volatilità locale, entrambi punti di riferimento e con la caratteristica fondamentale di garantire l'assenza di arbitraggio: il modello di Andreasen e Huge (e relative modifiche proposte da Abasto, Hientzsch e Kust) dove si assume non servano ipotesi a priori sulle caratteristiche della superficie, ed il modello di Gatheral e Jacquier nel quale si utilizza invece una forma parametrica.

Una volta spiegati ed implementati, i modelli saranno confrontati sia per quanto riguarda la bontà nel riprezzare le opzioni *vanilla* scaricate dal mercato sia nel prezzare opzioni *path dependent* (per le quali la volatilità locale viene infatti adoperata nella pratica), così da poter avere un modello sul quale basarsi qualora si voglia utilizzare la volatilità locale garantendosi l'assenza di arbitraggio.

Indice

Elenco delle figure

Elenco delle tabelle

Elenco degli algoritmi

Introduzione

Un derivato è un contratto finanziario il cui valore dipende da un asset sottostante che può essere un titolo azionario, un tasso d'interesse, una valuta o un qualunque altro strumento finanziario. Tra i derivati più utilizzati troviamo le *opzioni*, definite come come strumenti finanziari mediante i quali l'acquirente assume la facoltà, ma non l'obbligo, di esercitare un diritto di acquisto o di vendita dell'attività sottostante ad un prezzo e ad una data prefissate.

Il modello matematico di Black&Scholes [7] per il calcolo dei prezzi delle opzioni è stato introdotto nel 1973 e al giorno d'oggi rappresenta un riferimento universalmente riconosciuto e largamente utilizzato per ottenere questi prezzi. Esso si basa sull'idea che il prodotto sottostante S evolva secondo la formula del moto browniano geometrico con coefficienti μ e σ costanti

$$dS_t = \mu S_t dt + \sigma S_t dW_t$$

e che il prezzo di un'opzione, per esempio *Call europea*, sia la soluzione di una precisa equazione a derivate parziali. Tale soluzione è della forma

$$C_t = S_t N(d_1) - K e^{-r(T-t)} N(d_2)$$

dove T è la scadenza del contratto, K il prezzo di esercizio e S_t il valore del sottostante al tempo t. (Si rimanda al Capitolo 1 per il significato dei altri parametri della formula).

Tale formula si basa su numerose ipotesi che, nella realtà, raramente sono soddisfatte. La violazione di alcune delle ipotesi alla base del modello porta quest'ultimo a commettere sistematici errori, se confrontato coi prezzi di mercato. La principale ipotesi puntualmente confutata è che la volatilità σ rimanga costante per ogni opzione al variare della scadenza e del prezzo di esercizio. Invertendo la formula del calcolo dei prezzi si può osservare che la volatilità estratta dai prezzi di mercato, chiamata *volatilità implicita*, non è costante ma ha una forma a U chiamata Smile e varia anche nel tempo.

Per assecondare l'evidenza empirica del fatto che la volatilità non sia costante ed avere uno strumento per poterla gestire sono stati proposti numerosi modelli negli ultimi vent'anni, sviluppando e generalizzando il lavoro iniziale di Black&Scholes.

Tra essi sono presenti anche i modelli a volatilità locale. Introdotti per la prima volta nel 1994 da Dupire [16] e Derman e Kani [14] questi modelli assumono che il coefficiente diffusivo del moto del sottostante non sia più costante ma sia una funzione deterministica del tempo e del valore assunto dal sottostante stesso:

$$dS_{\mathrm{t}} = rS_{\mathrm{t}} + \sigma_{\mathrm{LV}}(S_{\mathrm{t}}, t)S_{\mathrm{t}}\, dW_{\mathrm{t}}.$$

Questo modello ha la caratteristica principale di produrre dei prezzi realistici, molto simili a quelli presenti sul mercato, seppur non spiegando il motivo della forma a U della volatilità di essi. Diventa quindi di fondamentale importanza qualora si voglia calcolare il prezzo di prodotti più complessi, strutturati, i quali si basano sul prezzo di altre opzioni: più queste ultime sono in linea col mercato più il prodotto complesso avrà un prezzo coerente.

Ovviamente il modello a volatilità locale presenta anche lati negativi, primo tra i quali il fatto che per poter calcolare i prezzi delle opzioni esso richieda tutti i valori della volatilità implicita per ogni scadenza e strike, ma ovviamente non tutti sono presenti e consultabili sul mercato, rendendo il modello non immediatamente utilizzabile.

Per ovviare al problema negli ultimi anni sono state proposte diverse tecniche per poterlo aggirare ed ottenere l'intera superficie di volatilità, così da poter utilizzare la formula di Dupire; tra queste spiccano quelle che, oltre a risolvere il problema della mancanza di alcuni valori, producono dei prezzi che garantiscano l'assenza di arbitraggio (cioè la certezza di ottenere profitto senza prendere rischi, sfruttando la differenza di prezzo tra prodotti che non dovrebbero averla).

Scopo della tesi e la sua struttura

In questa tesi verranno confrontate due tecniche importanti, molto diverse tra loro come approccio al problema ma col medesimo scopo finale, con le quali ottenere tutta la superficie di volatilità necessaria per l'utilizzo dei modelli a volatilità locale, evitando la possibilità di creare arbitraggio. Poiché questi modelli a volatilità locale avranno il fine ultimo di prezzare opzioni più complesse, per esempio asiatiche, il confronto verrà concluso analizzando

quale tra di essi conviene usare nella pratica qualora si presenti la necessità di avere il prezzo di queste opzioni più complesse.

In particolare la tesi verrà articolata secondo la seguente struttura:

Volatilità implicita, dove viene introdotta più accuratamente la volatilità implicita e le sue caratteristiche. Vengono inoltre illustrati i problemi che si generano dovuti alla sua presenza nei prezzi di mercato.

Volatilità locale, dove si espone il lavoro di Dupire [16] e si analizzano gli aspetti pratici dell'utilizzo della volatilità locale.

Forme di arbitraggio, riguarda il problema della presenza dell'arbitraggio, dandone esempi pratici e sottolineando il fatto che è di primaria importanza garantirne l'assenza quando si calcolano i prezzi dei derivati.

Il modello Andreasen-Huge, è il capitolo riguardante il modello esposto da Andreasen e Huge [4]; verranno discusse le proprietà e verrà implementato, commentando l'algoritmo. Verranno poi esposte anche le modifiche presentate da Abasto, Hientzsch e Kust [1] e valutate alla luce della bontà nel riprezzare opzioni già presenti sul mercato.

Il modello Gatheral-jacquier, è il capitolo riguardante il modello parametrico di Gatheral e Jacquier [21]. Esso nasce per calibrare la superficie di volatilità implicita ma in questa tesi verrà adattato per ottenere poi quella locale, valutando anche di esso la bontà nel riprezzare tramite Dupire le opzioni di mercato.

Replica di opzioni path dependent, dove vengono calcolate con i modelli esposti alcune opzioni path dependent come opzioni asiatiche e con barriera, per poter decretare quale modello sia meglio utilizzare a questo scopo.

Conclusioni, capitolo nel quale si fa il punto della situazione commentando i risultati ottenuti e gli obbiettivi raggiunti.

Nelle appendici sono presenti i dati scaricati da Bloomberg necessari per le calibrazioni ed i confronti, i metodi numerici utilizzati ed i passaggi per ottenere alcune formule. Sono presenti in ultimo anche i codici MATLAB tramite i quali è stata sviluppata questa tesi.

Capitolo 1

Volatilità implicita

La volatilità è quella grandezza che esprime l'ampiezza delle variazioni subite dal prezzo di un titolo. Definisce dunque la minore o maggiore facilità con cui tale prezzo varia al variare del rendimento richiesto. Ci sono diverse linee di pensiero su quali siano i reali motivi che la generano, ma quella più largamente condivisa è che essa è causata da nuove informazioni che raggiungono il mercato, a seguito delle quali i trader e i practitioners basato le loro valutazioni per l'acquisto o la vendita dei prodotti.

In questo capitolo verrà spiegato cosa è la volatilità implicita, in che modo viene utilizzata e perché è così importante riuscire a modellizzarla in maniera accurata.

1.1 Da Black&Scholes alla volatilità implicita

La formula di Black&Scholes [7] è stata utilizzata dalla sua creazione fino ad oggi su larghissima scala per valutare il prezzo di opzioni **plain vanilla Call e Put**, cioè i due prodotti derivati più semplici esistenti e usati come "mattoni" per tutti gli altri strumenti più complessi e strutturati.

Riassumendo brevemente come ottenerla, senza andare troppo nel dettaglio sui passaggi in quanto esula dallo scopo della tesi, si può partire definendo un moto browniano geometrico

$$\frac{dS}{S} = \mu dt + \sigma dW \tag{1.1}$$

con coefficienti μ e σ costanti. Nel caso più semplice di un opzione Call, l'equazione a derivate parziali che dà come risultato il prezzo $C(S_0, 0)$ è

$$\frac{\partial C}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 C}{\partial S^2} + rS\frac{\partial C}{\partial S} - rC = 0 \quad su[0,T) \times \mathbb{R}^+$$

con la condizione finale

$$C(S_{\mathrm{T}}, T) = \max(S_{\mathrm{T}} - K, 0).$$

Il problema può essere risolto *analiticamente* e la soluzione è della forma

$$C(S_{\mathrm{t}}, t) = S_{\mathrm{t}} N(d_1) - K e^{-r(T-t)} N(d_2) \tag{1.2}$$

$$P(S_{\mathrm{t}}, t) = -S_{\mathrm{t}} N(-d_1) + K e^{-r(T-t)} N(-d_2) \tag{1.3}$$

con:

$$d_1 = \frac{log(S_{\mathrm{t}}/K) + (r + \sigma^2/2)(T-t)}{\sigma\sqrt{T-t}}, \quad d_2 = d_1 - \sigma\sqrt{T-t} \tag{1.4}$$

dove C si riferisce al prezzo della Call, P alla Put e le altre grandezze delle formule sono:

N(x) la funzione di ripartizione della Normale Standard,

$\boldsymbol{S_t}$ il prezzo dello strumento finanziario sottostante al derivato al tempo t,

$\boldsymbol{K}$ lo strike price o prezzo di esercizio,

$\boldsymbol{r}$ il tasso di interesse privo di rischio,

$\boldsymbol{T}$ la scadenza del contratto in frazioni d'anno,

σ la deviazione standard percentuale istantanea del logaritmo del prezzo del titolo sottostante, espressa in termini annui.

Prendendo per esempio l'opzione Call, la notazione corretta da utilizzare sarebbe $C_{\mathrm{BS}}(S, t, T, K, r, \sigma)$. Tutte le voci di questo elenco sono valori deterministici e direttamente osservabili, tranne la volatilità (σ) che deve essere stimata studiando lo storico dei valori (e in questo caso prende il nome di volatilità storica, predecessore della volatilità stocastica) oppure estratta dal prezzo delle opzioni (volatilità implicita, in quanto è un valore implicito nel prezzo del derivato di riferimento). La volatilità storica riassume il comportamento passato della volatilità, tuttavia essa di fatto non è per nulla costante, ma varia, e di parecchio, nel tempo. Nei mercati purtroppo l'andamento futuro dipende pesantemente da eventi futuri che oggi sfuggono al nostro controllo e da informazioni che oggi non si conoscono, quindi è stato introdotto il modo per stimarla in accordo con le *aspettative future di mercato* sulla volatilità, da qui la cosidetta **volatilità implicita** [26].

> Whereas historical volatilities are backward looking, implied volatility are Forward looking.
>
> (Hull, 2008, [26]).

La volatilità implicita è definita come quel valore $\tilde{\sigma}$ che sostituito nell'equazione di Black&Scholes, a parità di tutti gli altri parametri, riproduce il prezzo di un opzione presente sul mercato, cioè definendo C il prezzo che viene calcolato e $\tilde{C}$ quello di mercato,

$$\tilde{\sigma} = \{\sigma \mid C(\sigma, \dots) = \tilde{C}\}. \tag{1.5}$$

E' possibile invertire l'equazione grazie alla monotonia del prezzo rispetto al volatilità, cioè la *Vega* dell'opzione

$$\nu(S,t) = \frac{\partial C}{\partial \sigma}(S, t, T, K, r, \sigma)$$

che non cambia segno: quindi un qualsiasi algoritmo che trova lo zero di una funzione, come Newton-Raphson o bisezione, può essere utilizzato per questo fine.

> *Implied volatility is the wrong number to put in the wrong formula to obtain the right price.*
>
> (Rebonato, 2005, [35]).

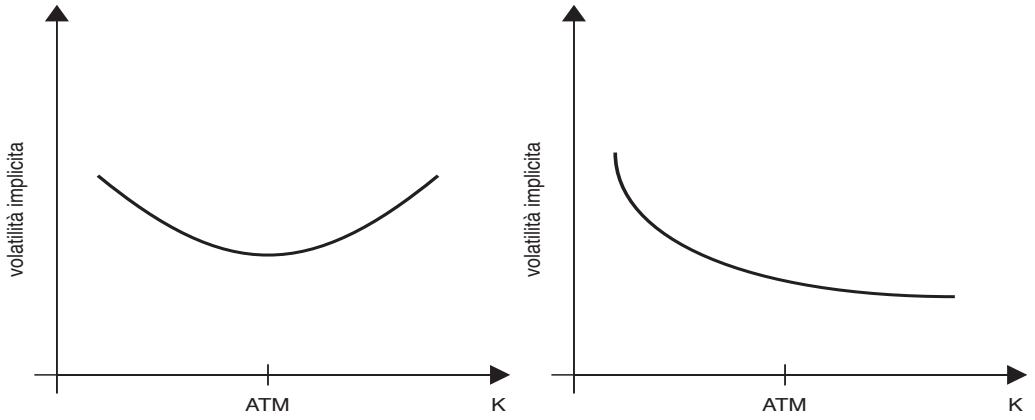
1.2 Smile e Skew: l'irregolarità della superficie

In teoria, Supponendo che i prezzi delle opzioni esistenti siano in accordo con la formula di Black&Scholes[1], la soluzione σ non dovrebbe variare per i diversi valori di K e di T, cioè dello strike e della scadenza dell'opzione, in quanto dovrebbe dipendere solo dal sottostante. In pratica invertendo i prezzi si può osservare il fenomeno detto *volatility smile*, nel quale la sigma, per un t fissato ed al variare degli strike K, è descritta come una curva a U con minimo nell'intorno delle opzioni at-the-money ($S_t = K$) per poi alzarsi

[1] Ipotesi forte ma non troppo azzardata, perché nonostante ci siano molte evidenze empiriche e teoriche della sua fallacia, nella pratica è comunque molto usata per la sua semplicità di comprensione, e per mantenere una certa conformità dove storicamente è sempre stata usata.

man mano che ci si sposta verso valori in-the-money ($S_t > K$) o out-of-money ($S_t < K$). Inoltre i due lati della curva possono presentare pendenze diverse, creando una certa asimmetria, la Skew, accentuata o meno a seconda del mercato considerato (questo effetto quindi è nascosto all'apparenza, ma compare una volta invertita la formula, come il sorriso del gatto del Cheshire in "Alice nel Paese delle Meraviglie").

Ovviamente la volatilità implicita non è costante neanche rispetto al tempo, ma descrive una curva che può essere sia decrescente sia crescente a seconda del sentimento degli investitori circa la rischiosità di un periodo rispetto a un altro, e con pendenze anch'esse variabili tra periodi e mercati. La curva nel tempo corrispondente al caso at-the-money prende il nome di Struttura a Termine, ed è il riferimento principale relativamente al prezzo per capire cosa il mercato si aspetta per il futuro di quel particolare sottostante [36] e [26]. A seconda del tipo di opzione considerata, che usi come sottostante un titolo azionario, una commodity o un tasso di interesse, questa superficie in K e T assumerà valori e pendenze diverse, rendendolo un argomento di studio tanto interessante quanto complesso e articolato, e portando alla creazione di nuove teorie e modelli (o nuovi parametri per quelli già esistenti) per poter spiegare una caratteristica piuttosto che un'altra della superficie [11].



(a) volatilità implicita tipica del mercato forex

(b) volatilità implicita tipica del mercato equity

Figura 1.1: Esempi di volatilità non costante nello strike.

A titolo di esempio di osservi la Figura 1.2 della superficie di volatilità implicita del titolo S&P500 al variare degli strike e delle maturity. Verrà spiegato nell'Appendice A dedicata ai dati raccolti dal mercato perché assume quella particolare forma, per il momento ci basta notare la pendenza della superficie, con molta asimmetria (con valori alti per strike più bassi del at-

the-money) e con Smile via via più accentuato per scadenze ravvicinate. Se il sottostante fosse stato per esempio un tasso di cambio, avremmo notato molta più simmetria tra le due ali, e molta più pendenza, cioè Skew minore e Smile più marcato.

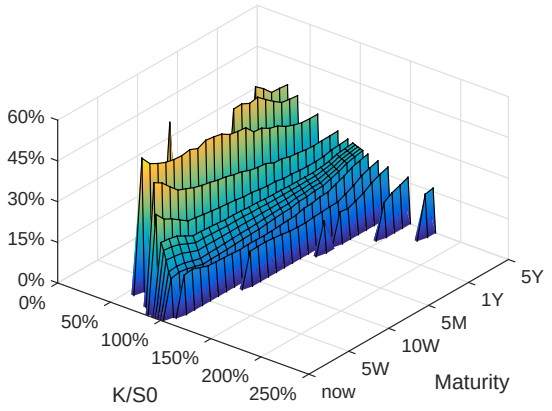


Figura 1.2: Superficie della volatilità implicita con i soli valori dal mercato.

Una domanda sorge a questo punto: quali sono le cause che portano alla formazione di una superficie di volatilità diversa da quella *flat* teorizzata nel modello Black&Scholes? La risposta è tanto semplice quanto difficile da gestire, la legge della domanda e dell'offerta: i trader e i Market Maker espongono sempre i prezzi ai quali sono disposti a comprare (ask) o a vendere (bid) determinate opzioni e ognuno ha prezzi leggermente diversi dovuti a molteplici fattori, in primis il sentimento riguardo a come evolverà il mercato. Le idee diverse sul futuro fanno sì che i diversi attori preferiscano comprare e vendere a prezzi diversi da quello teorico, modificando la curva dei prezzi esposti sul mercato.

Capitolo 2

Volatilità locale

Come esposto nel capitolo precedente, se il modello di Black&Scholes [7] fosse perfetto, il valore implicito della volatilità dovrebbe essere lo stesso per ogni opzione scritta sullo stesso sottostante, ma nel mondo reale non è questo il caso. Nel 1994 Bruno Dupire, discostandosi dalla classica equazione di Black&Scholes, introdusse una nuova equazione per prezzare le opzioni ed espose un nuovo modello per studiare il problema della volatilità implicita chiamato modello a **volatilità locale** [16]. Nella prima parte del capitolo verrà spiegata tale procedura, mentre nella seconda verranno analizzati i pro e i contro di questa teoria alternativa, alla luce degli altri modelli per la cattura della volatilità esistenti, in primis quello con volatilità stocastica.

2.1 L'equazione di Dupire

Il contributo sostanziale del modello di Dupire sta nel fatto di aver considerato la volatilità come un funzione deterministica non costante nel tempo e nel valore del sottostante. Definita la funzione $(S,t) \mapsto \sigma(S,t)$ la superficie di volatilità locale, si può considerare la seguente evoluzione del sottostante:

$$dS_{\mathrm{t}} = rS_{\mathrm{t}} + \sigma_{\mathrm{LV}}(S_{\mathrm{t}},t)S_{\mathrm{t}}\ dW_{\mathrm{t}}. \tag{2.1}$$

La corrispettiva equazione per il prezzo di un opzione su questo sottostante può essere vista quindi come una generalizzazione della classica formula di Black&Scholes, nella quale la volatilità non è costante. Usando quest'ultima però non è facile estrarre la σ_{LV} dai dati di mercato, quindi Dupire ha introdotto un'equazione diversa, che permette di ottenere il valore della σ_{LV} in forma chiusa e garantendo l'unicità della soluzione.

L'equazione parabolica all'indietro di Black&Scholes nelle variabili (S,t) infatti è la rappresentazione di Feynman-Kaç del valore atteso attualizzato

del valore futuro dell'opzione. E' possibile ottenere lo stesso prezzo risolvendo il problema duale, cioè un'equazione parabolica in avanti nelle variabili (K, t) che ha preso il nome di **equazione di Dupire**. Questa equazione non è nient'altro che l'equazione di Fokker-Planck (o Kolmogorov in avanti) per la funzione di densità di probabilità per il sottostante integrata due volte.

La formula di Dupire ha il vantaggio immediato che, invece di doverla risolvere ogni volta per ogni strike e maturity come nel caso Black&Scholes, fornisce già da subito tutti i prezzi dell'opzione considerata per ogni strike e maturity, dandogli in ingresso il valore spot del sottostante e il tempo.

Proposizione 1 (Equazione di Dupire [16])**.** *Il valore di una opzione Call come funzione dello strike K e della maturità T fissato il valore del sottostante S_0 è dato dall'equazione parabolica in avanti:*

$$\begin{cases} \frac{\partial C}{\partial T} - \frac{1}{2}\sigma_{\mathrm{LV}}^2(K,t)K^2\frac{\partial^2 C}{\partial K^2} + rK\frac{\partial C}{\partial K} = 0, & \\ C(0,t) = S_{\mathrm{t}}, & \forall t \in (0,\infty) \\ \lim_{K\to+\infty} C(K,t) = 0, & \forall t \in (0,\infty) \\ C(K,0) = \max(0, S_0 - K), & \forall K \in (0,\infty) \end{cases} \tag{2.2}$$

Dimostrazione. Definendo la funzione densità risk-neutral $\varphi(S_0, t, S_{\mathrm{T}}, T)$, il prezzo di un opzione Call può essere espresso come

$$\begin{aligned} C(S_0,0,K,T) &= e^{-rT}\int_0^\infty \varphi(S_0,0,S_{\mathrm{T}},T)C(S_{\mathrm{T}},T,K,T)\ dS_{\mathrm{T}} \\ &= e^{-rT}\int_K^\infty \varphi(S_0,0,S_{\mathrm{T}},T)(S_{\mathrm{T}}-K)\ dS_{\mathrm{T}} \end{aligned} \tag{2.3}$$

dalla quale si possono ottenere

$$\frac{\partial C}{\partial K} = -e^{-rT}\int_K^\infty \varphi(S_0,0,S_{\mathrm{T}},T)\ dS_{\mathrm{T}}, \tag{2.4a}$$

$$\frac{\partial^2 C}{\partial K^2} = e^{-rT}\varphi(S_0,0,K,T), \tag{2.4b}$$

$$\frac{\partial C}{\partial T} = -rC + e^{-rT}\int_K^\infty (S_{\mathrm{T}}-K)\frac{\partial \varphi}{\partial T}\ dS_{\mathrm{T}} \tag{2.4c}$$

dove φ è governato dall'equazione di Kolmogorov

$$\frac{\partial \varphi}{\partial T} = \frac{1}{2}\frac{\partial^2}{\partial S_{\mathrm{T}}^2}[\sigma(S_{\mathrm{T}},T)^2 S_{\mathrm{T}}^2\varphi(S_{\mathrm{T}},T)] - \frac{\partial}{\partial S_{\mathrm{T}}}[rS_{\mathrm{T}}\varphi(S_{\mathrm{T}},T)]. \tag{2.5}$$

Risolvendo l'equazione (2.4c) per parti due volte e inserendoci le (2.4a) e (2.4b) si arriva a

$$\begin{aligned}\frac{\partial C}{\partial T} &= -rC + e^{-rT}\Big(\frac{1}{2}\sigma^2 K^2 \varphi(S_0, 0, K, T) + r\int_K^\infty S_{\mathrm{T}}\varphi(S_0, 0, S_{\mathrm{T}}, T)\ dS_{\mathrm{T}}\Big) \\ &= -rK\frac{\partial C}{\partial K} + e^{-rT}\frac{1}{2}\sigma^2 K^2\varphi \end{aligned} \tag{2.6}$$

dalla quale si ottiene

$$\frac{\partial C}{\partial T} = \frac{1}{2}\sigma^2_{\mathrm{LV}} K^2 \frac{\partial^2 C}{\partial K^2} - rK\frac{\partial C}{\partial K} \tag{2.7}$$

cioè l'equazione di Dupire cercata. Per una dimostrazione più rigorosa guardare [17] e [36].

□

Osservazione 1. *Dupire suppone senza perdere troppo in generalità che il tasso di interesse privo di rischio sia zero; anche qui e nell'intera tesi verrà considerato tale, come del resto è approssimativamente la situazione attuale.*

Proposizione 2 (Formula di Dupire)**.** *Grazie alla forma dell'equazione* (2.2)*, è possibile invertirla ottenendo una formula esplicita per il calcolo della volatilità*

$$\sigma^2_{\mathrm{LV}} = \frac{(\partial C/\partial T) + rK(\partial C/\partial K)}{\frac{1}{2}K^2(\partial^2 C/\partial K^2)}\ . \tag{2.8}$$

Assumeremo però che esistano i prezzi di mercato per ogni Strike e per ogni maturity nel continuo, così da poterne creare una superficie (la superficie di volatilità) al variare di questi due parametri [32].

2.1.1 Volatilità locale espressa come valore atteso

Al posto di considerare l'equazione in avanti di Kolmogorov, è possibile utilizzare un differente approccio presente in [41] partendo dall'equazione (2.3). Riformulandola possiamo scriverla come

$$\begin{aligned} C &= \mathbb{E}[e^{-rT}(S_{\mathrm{T}} - K)^+ | \mathcal{F}_0] \\ &= \mathbb{E}[e^{-rT}(S_{\mathrm{T}} - K)\mathbf{1}_{\{S_{\mathrm{T}} > K\}} | \mathcal{F}_0] \end{aligned} \tag{2.9}$$

dove $\mathbf{1}_{\{\cdot\}}$ è la funzione indicatrice la cui derivata rispetto a S è la funzione delta di Dirac $\delta(\cdot)$.

Considerando le normali assunzioni di integrabilità vale il teorema di Fubini e possono essere scambiati l'operatore del valore atteso con la derivata, ottenendo così

$$\begin{aligned}\frac{\partial C}{\partial K} &= \frac{\partial}{\partial K}\mathbb{E}[e^{-rT}(S_{\mathrm{T}}-K)\mathbf{1}_{\{S_{\mathrm{T}}>K\}}|\mathcal{F}_0] \\ &= -\mathbb{E}[e^{-rT}\mathbf{1}_{\{S_{\mathrm{T}}>K\}}|\mathcal{F}_0] - \mathbb{E}[e^{-rT}(S_{\mathrm{T}}-K)\delta(S_{\mathrm{T}}-K)|\mathcal{F}_0] \\ &= -\mathbb{E}[e^{-rT}\mathbf{1}_{\{S_{\mathrm{T}}>K\}}|\mathcal{F}_0] \end{aligned} \tag{2.10}$$

$$\begin{aligned}\frac{\partial^2 C}{\partial K^2} &= -\mathbb{E}[e^{-rT}\frac{\partial}{\partial K}\mathbf{1}_{\{S_{\mathrm{T}}>K\}}|\mathcal{F}_0] \\ &= \mathbb{E}[e^{-rT}\delta(S_{\mathrm{T}}-K)|\mathcal{F}_0] \end{aligned} \tag{2.11}$$

dalla quale si può vedere la funzione di densità della probabilità del prezzo del sottostante (uguale a K a scadenza) come il valore atteso della funzione delta di Dirac:

$$\varphi(T,K) = \mathbb{E}[\delta(S_{\mathrm{T}}-K)|\mathcal{F}_0]\ .$$

Applicando la formula di Ito all'equazione (2.9)

$$\begin{aligned} dC &= \mathbb{E}\big[d\big(e^{-rT}(S_{\mathrm{T}}-K)\mathbf{1}_{\{S_{\mathrm{T}}>K\}}\big)|\mathcal{F}_0\big] \\ &= \mathbb{E}\Big[\frac{\partial e^{-rT}}{\partial T}(S_{\mathrm{T}}-K)\mathbf{1}_{\{S_{\mathrm{T}}>K\}}dT + e^{-rT}\frac{\partial}{\partial s}[(s-K)\mathbf{1}_{\{s>K\}}]\Big|_{s=S_{\mathrm{T}}}dS_{\mathrm{T}} \\ &+ e^{-rT}\frac{1}{2}\frac{\partial^2}{\partial s^2}[(s-K)\mathbf{1}_{\{s>K\}}]\Big|_{s=S_{\mathrm{T}}}dS_{\mathrm{T}}\cdot dS_{\mathrm{T}}|\mathcal{F}_0\Big] \end{aligned} \tag{2.12}$$

e utilizzando le seguenti identità

$$\begin{aligned}\frac{\partial}{\partial s}[(s-K)\mathbf{1}_{\{s>K\}}] &= \mathbf{1}_{\{s>K\}} + (s-K)\delta(s-K) = \mathbf{1}_{\{s>K\}} \\ \frac{\partial^2}{\partial s^2}[(s-K)\mathbf{1}_{\{s>K\}}] &= \frac{\partial}{\partial s}\mathbf{1}_{\{s>K\}} = \delta(s-K)\end{aligned}$$

l'espressione risultante per la (2.12) è

$$\begin{aligned} dC &= e^{-rT}\mathbb{E}\Big[-r_{\mathrm{T}}(S_{\mathrm{T}}-K)\mathbf{1}_{\{S_{\mathrm{T}}>K\}}dT + \mathbf{1}_{\{S_{\mathrm{T}}>K\}}S_{\mathrm{T}}[r_{\mathrm{T}}dT+\sigma(T,S_{\mathrm{T}})dW_{\mathrm{T}}] \\ &+ \frac{1}{2}\delta(S_{\mathrm{T}}-K)S_{\mathrm{T}}^2\sigma^2(T,S_{\mathrm{T}})dT|\mathcal{F}_0\Big] \\ &= e^{-rT}\mathbb{E}\big[r_{\mathrm{T}}K\mathbf{1}_{\{S_{\mathrm{T}}>K\}} + \frac{1}{2}\delta(S_{\mathrm{T}}-K)K^2\sigma^2(T,S_{\mathrm{T}})|\mathcal{F}_0\big]dT \end{aligned} \tag{2.13}$$

$$\Rightarrow \frac{\partial C}{\partial T} = r_{\mathrm{T}}e^{-rT}K\mathbb{E}[\mathbf{1}_{\{S_{\mathrm{T}}>K\}}|\mathcal{F}_0] + \frac{1}{2}e^{-rT}K^2\mathbb{E}[\delta(S_{\mathrm{T}}-K)\sigma^2(T,S_{\mathrm{T}})|\mathcal{F}_0] \tag{2.14}$$

l'ultimo termine della derivata può essere riscritto come

$$\mathbb{E}[\delta(S_{\mathrm{T}} - K)\sigma^2(T, S_{\mathrm{T}})|\mathcal{F}_0] = \mathbb{E}[\sigma^2(T, S_{\mathrm{T}})|S_{\mathrm{T}} = K, \mathcal{F}_0] \cdot \mathbb{E}[\delta(S_{\mathrm{T}} - K)|\mathcal{F}_0].$$

Utilizzando ora la (2.10), si ottiene

$$\frac{\partial C}{\partial T} = -r_{\mathrm{T}}\frac{\partial C}{\partial K} + \frac{1}{2}K^2\mathbb{E}[\sigma^2(T, S_{\mathrm{T}})|S_{\mathrm{T}} = K, \mathcal{F}_0]\frac{\partial^2 C}{\partial K^2} \tag{2.15}$$

$$\Rightarrow \mathbb{E}[\sigma^2(T, S_{\mathrm{T}})|S_{\mathrm{T}} = K, \mathcal{F}_0] = 2\frac{\frac{\partial C}{\partial T} + r_{\mathrm{T}}\frac{\partial C}{\partial K}}{K^2\frac{\partial^2 C}{\partial K^2}} \ . \tag{2.16}$$

Confrontando l'equazione (2.16) con la (2.8) si nota che la varianza locale può essere vista come la varianza attesa a maturity dato il prezzo del sottostante uguale al prezzo strike. Questo modo di vedere la volatilità locale come la media delle volatilità istantanee formalizza d'altronde l'intuizione di quei trader e practitioner che per primi introdussero questo concetto. Si legge infatti in [19]:
-It is unlikely that Dupire, Derman, and Kani ever thought of local volatility as representing a model of how volatilities actually evolve. Rather, it is likely that they thought of local volatilities as representing some kind of average over all possible instantaneous volatilities in a stochastic volatility world (an "effective theory"). Local volatility models do not therefore really represent a separate class of models; the idea is more to make a simplifying assumption that allows practitioners to price exotic options consistently with the known prices of vanilla options-.

2.2 Confronto con altre tecniche

Oltre al modello a volatilità locale, sono stati introdotti altri metodi per gestire il problema della volatilità implicita, tra i quali il più importante è l'utilizzo della volatilità stocastica diffusiva. In questo modello il prezzo (risk-neutral) del sottostante evolve secondo la regola:

$$\begin{cases} dS_{\mathrm{t}} = \mu S_{\mathrm{t}}\ dt + \sigma_{\mathrm{t}} S_{\mathrm{t}}\ dW_{\mathrm{t}} \\ \sigma_{\mathrm{t}} = f(y_{\mathrm{t}}) \end{cases} \tag{2.17}$$

dove σ_{t} è un altro processo stocastico, di solito con le caratteristiche di essere *mean-reverting* e sempre positivo[1]. Un esempio molto diffuso nella pratica è

[1]Quindi $f(y_{\mathrm{t}})$ è scelto in modo tale da non produrre mai volatilità negative e da avere una componente che fa riavvicinare il processo alla sua media di lungo periodo quando la componente stocastica lo fa allontanare.

il modello di Heston [25] nel quale

$$\begin{cases} f(y_t) = \sqrt{y_t} \\ dy_t = \lambda(\eta - y_t)\ dt + \theta\sqrt{y_t}\ dZ_t \\ \rho \neq 0 \quad \wedge \quad \lambda, \eta, \theta > 0 \end{cases} \tag{2.18}$$

dove Z_t è un altro processo di Wiener correlato con W_t con coefficiente di correlazione ρ. Questo processo è sempre positivo e continuo perché anche se toccasse zero, il termine diffusivo si annullerebbe e il drift rimanderebbe il processo nella direzione positiva[2].

Un altro modello utilizzato è quello a volatilità stocastica con salti, la cui versione più famosa è quella introdotta da Bates, dove unisce Heston al modello Jump-Diffusion di Merton, aggiungendo dei salti proporzionali alla distribuzione log-normale nel moto del sottostante. Questo fa sì che il modello risponda bene anche per scadenze molto ravvicinate e con più asimmetria. Per completezza va aggiunto che esistono anche i modelli in cui vengono utilizzati i processi con salto di Lèvy con subordinatori, dove il tempo non è lineare, ma varia con la frenesia del mercato e possono quindi rispondere meglio ai diversi scenari di volatilità che si possono verificare [11].

Questi modelli stocastici, avendo più parametri, hanno in generale la capacità di rispondere meglio ai problemi di smile e asimmetria della volatilità implicita, nonché dell'effetto clustering (inerzia nella serie storica) e code più spesse della normale (modellizzano meglio gli eventi rari).

Ovviamente sia il modello a volatilità stocastica che quello a volatilità locale però hanno anche aspetti negativi. Il più rilevante per quanto riguarda la volatilità stocastica è la presenza di una fonte di casualità in più (mettere i due W_t e Z_t per esempio) e del maggior numero di parametri da dover stimare, quindi computazionalmente più pesante. La caratteristica di avere più fonti di casualità ha come effetto non trascurabile il fatto di rendere il modello **incompleto**. Il concetto di completezza è strettamente collegato con il teorema di Girsanov e con l'esistenza di una misura di martingala equivalente. Infatti se il modello è completo allora esiste un'unica misura equivalente e il prezzo di ogni derivato è determinato univocamente. Di contro il modello a volatilità locale gode di questa proprietà, quindi è considerato **completo**. La perdita della completezza e l'introduzione di sorgenti di rischio non scambiabili sul mercato, fa sì che si perda quindi anche la possibilità di coprirsi dal rischio dell'investimento usando una precisa quantità statica del sottostante

[2]La stessa cosa avviene nel caso della volatilità locale, dove il processo non tocca mai zero se $y_0 > 0$.

oppure altri parametri legati alle *greche* del derivato [26] (strategia chiamata *hedging* in ambito finanziario).

Per quanto riguarda invece le critiche mosse alla volatilità locale, va preso in considerazione il fatto che ha delle pessime proprietà statistiche e previsive [15]: cambia infatti considerevolmente nel tempo (i parametri cambiano anche di molto da una settimana all'altra) ed è un modello creato *ad hoc*: non esiste una spiegazione economica alla funzione di volatilità locale e al valore che assume in relazione ai parametri. Nella pratica inoltre il fatto che possa essere usato come hedge statico, cioè fissato a $t = 0$ e non più modificato[3], ha poco valore in quanto si preferisce un hedge dinamico, che varia durante la vita del prodotto adattandosi alle diverse situazioni del mercato.

Riassumendo come scritto in [12], da un lato la volatilità locale può riprodurre esattamente i prezzi delle opzioni vanilla e replicare in maniera statica ma perfetta la volatilità implicita, dall'altro invece la volatilità stocastica commette più errori nel prezzo ma esibisce una dinamica più realistica della volatilità implicita. In ultima analisi un punto a favore della volatilità locale, che fa prediligere questa rispetto alla stocastica nello studio svolto in questa tesi, consiste nel fatto che, sotto opportune modifiche e ipotesi, è possibile rendere il modello privo di arbitraggio (tema esposto nel Capitolo 3).

2.3 Calibrazione

L'obbiettivo di calibrare accuratamente la funzione di volatilità locale con un numero limitato di osservazioni dal mercato - con un costo computazionale e di memoria il più limitato possibile - è stato analizzato in numerosi articoli. Le procedure risultanti possono essere divise in due approcci principali:

1. Calibrazione diretta della volatilità locale dai dati, dove devono essere fatte delle assunzioni sulla forma funzionale dei parametri della volatilità per poter garantire l'unicità della soluzione;

2. Fitting indiretto dei parametri per ottenere una superficie arbitrage free dei prezzi (o la corrispondente superficie implicita) consistente coi dati attraverso l'equazione (2.8). Poiché la volatilità locale dipende dalle derivate di questi prezzi dell'opzione, questa superficie deve essere *smooth* abbastanza per poter ottenere dei risultati affidabili. Diversi metodi sono stati implementati in questa direzione, e a loro volta possono essere divisi in due categorie: parametrici e non-parametrici.

[3]Stesso meccanismo della gestione dei portafogli chiamata *buy and hold* dove una volta fissati i pesi di ogni asset, non vengono più toccati fino a scadenza.

Un esempio di calibrazione diretta è il metodo presente nell'articolo [10]: Gli autori propongono di approssimare la funzione di volatilità locale tramite una *spline* cubica $c(t,s) \sim \sigma(t,s)$ poiché le spline sono spesso usate per approssimare curve e superfici smussate. Una semplice occhiata alla regione di interesse $S \times T : (0,\infty) \times (0, T_{\max})$ giustifica la scelta.

Un altro esempio di modello dove vengono fatte delle ipotesi sulla forma funzionale è quello proposto da Gatheral e Jacquier [21] chiamato Stochastic Volatility Inspired (SVI). Questo modello, spiegato più nel dettaglio nel Capitolo 5 dove ne verrà proposta e implementata una sua modifica per renderlo *arbitrage free*, prevede che per ogni *slice* temporale[4] la volatilità segua l'andamento descritto dalla curva (5.8); da essa si può poi ottenere qualsiasi valore al variare dello strike k. Dato l'esiguo numero di parametri da stimare e l'elevata accuratezza dei risultati, nonostante il problema della guess iniziale, il modello è considerato probabilmente il più famoso in quanto largamente citato ed utilizzato di questa categoria di metodi per la costruzione della superficie.

Per quanto riguarda l'approccio indiretto, il modello più importante di questa categoria è stato presentato in [4]. Come verrà spiegato nel dettaglio nel Capitolo 4, in questo modello l'Equazione alle derivate parziali (EDP) in avanti di Dupire è inizialmente discretizzata dal metodo implicito delle differenze finite e la funzione di volatilità locale è rimpiazzata da un'approssimazione secondo un metodo di ottimizzazione *full-scale*, cioè dove l'ottimizzazione non è fatta ogni volta su ogni singolo elemento ma a gruppi di essi o addirittura su tutti contemporaneamente. Questa volatilità approssimata è data da una funzione costante a tratti e time-independent univocamente determinata dai livelli di volatilità ϑ_{ij}. L'interpolazione tra le maturity osservate è successivamente effettuata risolvendo uno step temporale nel metodo implicito delle differenze finite.

Per completezza va aggiunto che il terzo modello analizzato in questa tesi, nella sezione 4.5, è uno sviluppo successivo del modello di Andreasen-Huge del Capitolo 4, pubblicato in [1]. La differenza più evidente dal suo predecessore è che viene fatta un'ottimizzazione chiamata *multi-step*, cioè direttamente tenendo conto dei possibili sotto intervalli tra le maturity presenti all'inizio nei dati di mercato ed ottimizzando differenti variabili.

[4] Il fatto di chiamarle *fette* (slice) temporali rende bene l'idea, perché osservando la superficie di volatilità, se si prendono tutti i valori per un t fissato, è equivalente a tagliare la superficie in corrispondenza di quel istante temporale. Poiché tutti i modelli presenti in questa tesi evolvono nel tempo, l'andamento segue i punti discreti t presenti, procedendo step-by-step, o alternativamente slice-by-slice.

Capitolo 3

Forme di Arbitraggio

In questo capitolo verrà esposto il concetto di arbitraggio e, contestualmente, della sua assenza e del perché è così importante nel mondo finanziario e nella creazione di modelli matematici coerenti ad esso collegati.[1]

L'assenza di opportunità di arbitraggio può essere "declinata" in diversi modi a seconda della necessità nel modello analizzato; si rimanda quindi ai capitoli successivi per una definizione di essa in linea col contesto nel quale verrà utilizzata, ovvero quali condizioni dovranno essere rispettate per far sì che i prezzi ottenuti risultino *arbitrage-free*.

3.1 Esempi di possibili arbitraggi

Il modo di dire (di derivazione inglese) più utilizzato per spiegare cosa sia un arbitraggio è *la capacità di ottenere il pranzo gratis*. Dandone una definizione più rigorosa [5]:

Definizione 1. *Il mercato offre opportunità di arbitraggio se è possibile costruire un portafoglio a costo nullo (o negativo) che garantisca un rendimento certo positivo (o non negativo) a scadenza; in altre parole se, in condizioni di incertezza, è possibile costruire in t=0 un portafoglio che soddisfi una delle seguenti due condizioni:*

1. *Sia a costo nullo e che garantisca un rendimento positivo in almeno uno stato del mondo e non negativo per tutti gli altri;*

2. *Sia a costo negativo e che garantisca un rendimento non negativo in ogni stato del mondo.*

[1] In logica matematica, una teoria formale si dice **coerente** (o non contraddittoria, talvolta anche consistente, per assonanza con l'inglese *consistent*) se in essa è impossibile dimostrare una contraddizione.

Un esempio di arbitraggio tramite un portafoglio è possibile considerando la presenza di un titolo privo di rischio (bond B), un titolo rischioso S e un'opzine Call europea C su sottostante S. Supponiamo che in $t = 0$ si abbia $B(0) = 1$, $S(0) = 1$ e che la Call abbia strike $K = 1$, scadenza T e quotazione $C(0) = 0.2$. Supponiamo inoltre che alla data di scadenta $t = T$ siano possibili solo due stati del mondo, *up* e *down*.

In $t = T$ nel caso si verifichi l'evento *up* il bond per esempio varrà $B(T) = 1.25$ e il sottostante $S(T, u) = 1.75$; l'opzione a quel punto pagherà $C(T, u) = 0.75$. Se si comprassero 9 bond e 20 Call e si vendessero 15 sottostanti il flusso di cassa sarebbe

	B	S	Call	totale
t=0	-9	15	-4	2
t=T	11.25	-26.25	15	0

Se si verificasse invece l'evento *down* il bond varrebbe comunque $B(T) = 1.25$ ma per esempio $S(T, d) = 0.75$ e quindi la Call $C(T, d) = 0$, con gli stessi pesi nel portafoglio si avrebbe

	B	S	Call	totale
t=0	-9	15	-4	2
t=T	11.25	-11.25	0	0

Esiste dunque possibilità di arbitraggio, ovvero l'investitore può ottenere un guadagno certo di 2 oggi senza esporsi ad alcun rischio domani. In questo caso la possibilità di arbitraggio è dovuta ad una quotazione errata della Call in $t = 0$; si capisce quindi fin da questo esempio quanto sia importante calcolare correttamente il prezzo delle opzioni così da evitare possibili arbitraggi.

Nella pratica il termine l'arbitraggio è riferito anche all'operazione che consiste nell'acquistare un bene o un'attività finanziaria su un mercato rivendendolo su un altro mercato, sfruttando le differenze di prezzo al fine di ottenere un profitto. Questo è possibile solo quando viene meno la *legge del prezzo unico* e l'atto di acquisto e di vendita devono essere attuati in brevissimo tempo, in teoria contemporaneamente, per evitare che una variazione del mercato possa annullare il guadagno.

Un esempio di questo tipo riguarda il tasso di cambio (strettamente correlato alla ipotetica differenza di prezzi): supponendo che il tasso di cambio nel mercato londinese sia $5£ = 8\$$ mentre quello a New York sia $8\$ = 6£$, se si convertissero $20£$ in $36\$$ e poi si riconvertissero i $36\$$ in $24£$, si avrebbe un profitto di $4£$ sfruttando la semplice differenza tra mercati.

3.2 Il principio di non arbitraggio

Dal punto di vista matematico questo concetto viene espresso in relazione al tasso di rendimento risk-free (cioè un tasso privo di alcun rischio, quale nella realtà potrebbe essere quello di un titolo di stato di una nazione solida come gli Stati Uniti o la Germania, oppure il tasso swap, cioè quel tasso fisso che fa sì che, al momento della stipula di un contratto Interest Rate Swap, i flussi di cassa della parte variabile *floating leg* eguaglino quelli della parte fissa *fixed leg*; quest'ultimo è molto più simile ad un vero tasso risk-free e quindi preferibile nella pratica [26]). Supponiamo che sul mercato esistano un titolo privo di rischio $G(t)$ ed un portafoglio di titoli auto-finanziato $V(t)$ che evolvano secondo

$$\begin{cases} dG(t) = r(t)G(t)\ dt \\ dV(t) = h(t,\omega)V(t)\ dt \end{cases} \tag{3.1}$$

pertanto, affinché non vi sia possibilità di arbitraggio, deve valere $h(t,\omega) = r(t)$ per qualsiasi t, dove $r(t)$ è il tasso di interesse istantaneamente privo di rischio[2] [6].

Tale proposizione equivale ad affermare che, se si riesce attraverso un'opportuna combinazione dei titoli presenti sul mercato ad ottenere un portafoglio che si evolve in modo deterministico, allora tale portafoglio deve rendere tanto quanto il titolo privo di rischio. Infatti se ad esempio $h(t,\omega)$ fosse maggiore di $r(t,\omega)$ allora tutti gli investitori venderebbero allo scoperto il titolo privo di rischio per acquistare il portafoglio $V(t)$. La vendita del titolo non rischioso ne determinerebbe una riduzione di valore tale da alzarne il rendimento e ricondurre il mercato all'equilibrio con $h(t,\omega) = r(t,\omega)$. Stessa situazione nel caso in cui $h(t,\omega)$ fosse minore di $r(t,\omega)$, dove si arriverebbe comunque all'equilibrio al passare del tempo per ragionamenti analoghi[3]. Un altro modo di vedere l'assenza di arbitraggio è rispetto all'asse temporale, in quanto può essere anche formalizzato come segue:

se due strumenti forniscono all'epoca finale $t = T$ lo stesso risultato, qualunque sia lo stato del mondo ω_j che si verifica, allora all'epoca iniziale $t = 0$ devono avere lo stesso prezzo.

[2] Il termine ω è lo stato di natura, cioè l'elemento dell'insieme degli eventi possibili Θ da cui si costruisce la sigma algebra ($\mathcal{F}$) con cui si misurano i processi di Wiener considerati sullo spazio di probabilità completo $(\Theta, \mathcal{F}, P)$.

[3] un'ulteriore ipotesi da fare è che il mercato non abbia frizioni, ossia non si considerano tasse, costi di transazione, è possibile prendere a prestito o prestare allo stesso tasso e sono permesse vendite allo scoperto.

La possibilità di poter confrontare quantità in istanti di tempo differenti sta alla base della seguente proposizione, la più importante conseguenza dell'assenza di arbitraggio e forse dopo il lemma di Ito il più importante concetto su cui si basa il pricing e tutta la matematica finanziaria:

Proposizione 3 (Valutazione Risk-Neutral)**.** *Il prezzo arbitrage-free al tempo t di un strumento finanziario $\Phi(S(T))$, con maturità T e sottostante $S(t)$ è dato dalla formula*

$$F(t,s) = e^{-rT}\mathbb{E}^{\mathrm{Q}}_{\mathrm{t,s}}[\Phi(S(T))] \tag{3.2}$$

Il valore atteso è preso rispetto alla misura di probabilità Q, cioè alla misura neutrale rispetto al rischio. Quest'ultima, chiamata più frequentemente misura di martingala, permette di interpretare il prezzo del titolo come valore atteso del suo payoff scontato tramite il tasso di interesse privo di rischio, dove il valore atteso è calcolato non rispetto alla probabilità reale P, ma rispetto ad un'altra che agisce come se tutti gli agenti sul mercato fossero *risk neutral*.

Il punto cardine è che l'assenza di arbitraggio ci garantisce che questa misura di probabilità equivalente Q sia unica, quindi produce **univocamente** il prezzo di qualsiasi strumento finanziario considerato.

Capitolo 4

Il modello Andreasen-Huge

I modelli a volatilità locale come Dupire [16], JPMorgan [29] e Andersen &Andreasen [2] idealmente richiedono una superficie di volatilità continua ovunque nello strike e nel tempo, corrispondente a prezzi di opzioni europee consistenti e **arbitrage-free** come input. Nella pratica, si possono osservare purtroppo solo un insieme discreto di prezzi delle opzioni (o equivalentemente dei valori della volatilità implicita).

Nel lavoro di Jesper Andreasen e Brian Huge [4] (chiamato nella tesi anche A-H), esposto in questo capitolo, viene spiegato un metodo per poter estrapolare da un insieme discreto di valori di volatilità implicite tutta la superficie di volatilità locali negli strike e nella maturity in modo che i prezzi corrispondenti siano privi di opportunità di arbitraggio. Una volta ottenuta, vengono usate le differenze finite per risolvere le equazioni a derivate parziali presenti (l'equazione di Dupire (2.2)), per ottenere così i prezzi delle opzioni cercate. Confrontandole poi con quelli presi dal mercato, è possibile vedere il grado di accuratezza delle stime del metodo.

Il capitolo si focalizzerà sulle tecniche di implementazione e relativi commenti e spiegazioni, mentre in Appendice E sono presenti i codici scritti in MATLAB® sviluppati e poi utilizzati per l'analisi che caratterizza questa tesi.

4.1 Spiegazione del modello

Data una griglia negli strike e nel tempo e un insieme di valori della volatilità implicita, è possibile riscrivere l'equazione (2.2) sostituendo l'operatore differenziale con l'operatore alle differenze, ottenendo il seguente schema alle

differenze finite:

$$\begin{gathered}[1-\frac{1}{2}\triangle t_{\mathrm{i}}\vartheta_{\mathrm{i}}(k)^2k^2\delta_{\mathrm{kk}}]C(t_{\mathrm{i+1}},k)=\ C(t_{\mathrm{i}},k),\\ C(0,k)=(S(t_0)-k)^+\end{gathered}\tag{4.1}$$

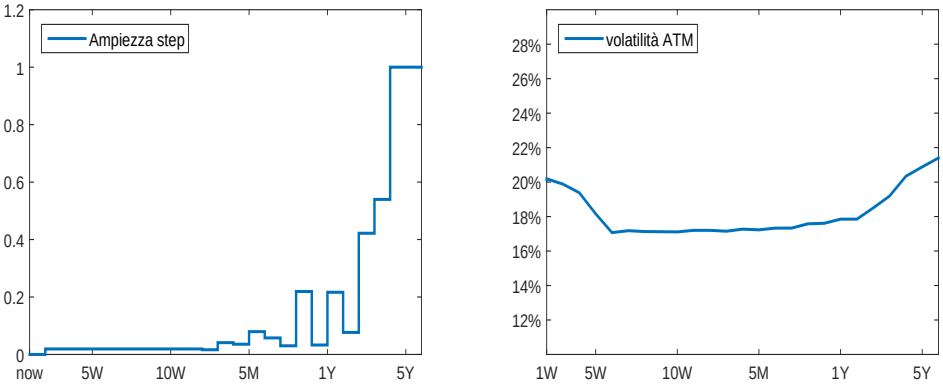
dove

$$\delta_{\mathrm{kk}}f(k)=\frac{1}{\triangle k^2}(f(k-\triangle k)-2f(k)+f(k+\triangle K))\quad i=0,1,\ldots$$

$$\triangle t_{\mathrm{i}}=t_{\mathrm{i+1}}-t_{\mathrm{i}}$$

$$k_{\mathrm{j}}=k_0+j\triangle k,\ \ j=0,1,\ldots,n.$$

Si noti che la discretizzazione del tempo, a differenza dello strike, non è equispaziata; questa modifica è necessaria in quanto nella realtà le scadenze dei prezzi sono molto più fitte nei periodi brevi e man mano si diradano allontanandosi dalla data di inizio: hanno un incremento che ricorda quello esponenziale ma, come si vede in Figura 4.1(a), sono presenti molte irregolarità.



(a) Intervallo temporale tra due scadenze espresso in frazioni d'anno

(b) Struttura a termine della volatilità At The Money

Figura 4.1: Proprietà dei dati scaricati nella dimensione Tempo.

Il problema (4.1) può essere risolto iterativamente nel tempo risolvendo un sistema matriciale con matrici tridiagonali, tipico delle differenze finite. Come verrà spiegato nella sezione in Appendice C, è possibile riscrivere il modello discretizzato di Dupire (4.1) come soluzione del sistema

$$AC(t_{\mathrm{i+1}})=C(t_{\mathrm{i}})\tag{4.2}$$

dove A è una matrice tridiagonale della forma:

$$A = \begin{bmatrix} 1 & 0 & & & & \\ -z_1 & 1+2z_1 & -z_1 & & & \\ & -z_2 & 1+2z_2 & -z_2 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -z_{n-1} & 1+2z_{n-1} & -z_{n-1} \\ & & & & 0 & 1 \end{bmatrix}$$

$$con \quad z_j = \frac{1}{2}\frac{\triangle t}{\triangle k^2}\vartheta_i(k_j)^2 k_j^2 \tag{4.3}$$

Per completezza vengono aggiunte anche le necessarie condizioni al bordo del sistema:

$$C_{kk}(t, k_0) = 0 \qquad C_{kk}(t, k_n) = 0. \tag{4.4}$$

Si noti che le condizioni (4.4) vengono già incorporate nella matrice (4.3) del modello: la prima e l'ultima riga di tale matrice infatti sono differenti rispetto alle altre ed hanno lo scopo di ricreare l'effetto di queste condizioni, e cioè quello di mantenere il prezzo per strike *estremi* costante, non influenzabile dalla diffusione: per valori lontani dallo strike ATM l'effetto della diffusione tende via via a scomparire e a rendere la superficie sempre più liscia e costante.

Nella matrice (4.3) sono presenti delle funzioni di volatilità ϑ_{ik} incognite[1], costanti a tratti sia nel tempo che nello spazio degli strike, che rappresentano il punto cardine del modello di Andreasen e Huge: dati i prezzi di mercato di una particolare opzione, bisogna trovare quali ϑ_{ik} minimizzino l'errore tra i prezzi generati da esse e i prezzi reali. Come verrà spiegato in seguito, questo passaggio verrà sviluppato tramite apposite funzioni di ottimizzazione vincolata non lineare presenti in MATLAB.

***Core* del modello** Il processo di determinazione delle volatilità viene fatto **step by step**: ad ogni istante di tempo i si risolve il problema di ottimizzazione minimizzando la differenza tra i prezzi ottenuti dal modello (C_i) e quelli presi dal mercato ($\widetilde{C}_i$) ricavando così le ϑ_{ik}, le quali vengono utilizzate mentre si risolve un passo del sistema alle differenze finite per ottenere i prezzi del modello(C_{i+1}); a questo punto si confrontano con i prezzi di mercato ($\widetilde{C}_{i+1}$) per determinare i $\vartheta_{i+1,k}$ e si itera il processo fino all'ultima scadenza.

[1]Quando ci si riferisce ad uno strike generico viene utilizzata la notazione ϑ_{ik} al posto di $\vartheta_i(k_j)$.

Così facendo costruiamo una griglia di prezzi negli strike e negli istanti temporali inizialmente considerati, ed è possibile ottenere il prezzo per qualsiasi $k \in]k_{\mathrm{j}}, k_{\mathrm{j+1}}[$ in quanto la funzione ϑ assume valore costante in quel tratto. Resta da risolvere però il problema delle scadenze non presenti esplicitamente nella griglia iniziale. Per ottenere questi prezzi assumendo un $t \in]t_{\mathrm{i}}, t_{\mathrm{i+1}}[$ bisogna quindi risolvere

$$C(t,k) = \int_0^\infty \frac{1}{t-t_{\mathrm{i}}} e^{-u/(t-t_{\mathrm{i}})} g(u,k)\, du, \quad t > t_{\mathrm{i}} \tag{4.5}$$

dove $g(u,k)$ è la soluzione di

$$\begin{gathered} 0 = -\frac{\partial g}{\partial u} + \frac{1}{2}\vartheta(k)^2 k^2 \frac{\partial^2 g}{\partial k^2}, \quad u > 0 \\ g(0,k) = c(t_{\mathrm{i}}, k). \end{gathered} \tag{4.6}$$

Questo non è nient'altro che il sistema (4.1) applicato usando un intervallo temporale di ampiezza $t - t_{\mathrm{i}}$ e con ϑ uguale al valore ϑ_{ik} (costante in tutto il tratto $t_{\mathrm{i}}, t_{\mathrm{i+1}}$). Così facendo si ha il notevole vantaggio che per ogni scadenza esclusa dalla griglia iniziale, si deve risolvere solo il prodotto matrice-vettore delle differenze finite, evitando il problema dell'ottimizzazione vincolata non lineare. Osservando la Figura 4.2, presente anche in [4], si può capire come lavora il codice: una volta stimati i valori in t_{i} e in $t_{\mathrm{i+1}}$, si parte da quelli per riempire gli spazi intermedi.

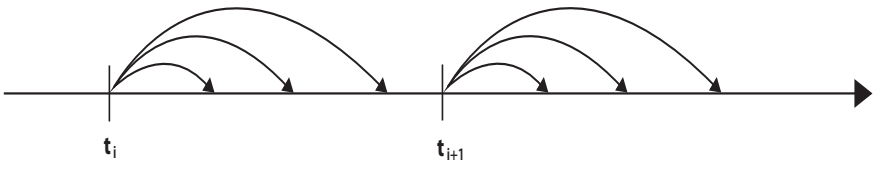


Figura 4.2: schema di avanzamento nel tempo del modello A-H.

4.2 Assenza di arbitraggio

La richiesta di assenza di arbitraggio del Capitolo 3 per i prezzi generati dal modello (sia quelli sulla griglia sia quelli generati dal riempimento per avere la continuità della superficie) viene formalizzata in questo contesto come due **condizioni sulle derivate** del prezzo che devono essere soddisfatte.

Proposizione 4 (Assenza di arbitraggio)**.** *Nel modello Andreasen-Huge è garantita l'assenza di arbitraggio se e solo se valgono le seguenti disuguaglianze:*

$$\begin{cases} C_{\mathrm{t}}(t,k) \geq 0 \\ C_{\mathrm{kk}}(t,k) \geq 0 \quad \forall (t,k) \end{cases} \tag{4.7}$$

In questo caso quindi si richiede la non negatività del prezzo derivato rispetto al tempo e la non negatività del prezzo derivato due volte rispetto allo strike.

Dimostrazione. Si considerino i prezzi delle opzioni generati dalla equazione in avanti di Dupire (2.2), risolta nel tempo t in avanti date le condizioni al bordo iniziali $g(0,k)$. Come mostrato in [3], l'equazione di Dupire può essere vista anche come un'equazione all'indietro per

$$G(t,k) = \mathbb{E}[g(0,k(0))|k(t) = K] \tag{4.8}$$

dove k segue il processo $dk(t) = \vartheta(t,k(t))dZ(t)$ e Z è un moto browniano backward. La mappa $g(0,\cdot) \mapsto g(t,\cdot)$ data dall'equazione (2.2) definisce così un funzionale positivo lineare nel senso che

$$g(0,\cdot) \geq 0 \rightarrow g(t,\cdot) \geq 0 \tag{4.9}$$

Inoltre, differenziando due volte la (2.2) rispetto ai k si ottiene l'equazione in avanti per $p = g_{\mathrm{kk}}$:

$$\begin{aligned} 0 &= -\frac{\partial p}{\partial t} + \frac{1}{2}\frac{\partial^2}{\partial k^2}[\vartheta(k)^2 p] \\ p(0,k) &= g_{\mathrm{kk}}(0,k) = \int g_{\mathrm{kk}}(0,l)\delta(k-l)\, dl. \end{aligned} \tag{4.10}$$

L'equazione (4.8) è equivalente alla Fokker-Planck per un processo generico $dx(t) = \vartheta(x(t))dW(t)$ dove W è il moto browniano standard. Questo fa sì che l'equazione di Dupire (2.2) preservi la convessità:

$$g_{\mathrm{kk}}(0,\cdot) \geq 0 \rightarrow g_{\mathrm{kk}}(t,\cdot) \geq 0. \tag{4.11}$$

Sia $T(u)$ una funzione strettamente crescente. Si definisce la trasformata (di Laplace) dei prezzi dell'opzione come

$$h(u,k) = \int_0^\infty \frac{1}{T(u)} e^{\frac{-t}{T(u)}} g(t,k)dt \tag{4.12}$$

Moltiplicando (2.2) per $e^{-t/T(u)}$ e integrando in t si ottiene

$$\Big[1 - \frac{1}{2}T(u)\vartheta(k)^2\frac{\partial^2}{\partial k^2}\Big]h(u,k) = g(0,k) \tag{4.13}$$

Dalla (4.9) e (4.11) concludiamo che (4.13) definisce un funzionale lineare positivo che preserva la convessità.

Differenziando (4.13) rispetto a u si arriva a

$$\Big[1 - \frac{1}{2}T(u)\vartheta(k)^2\frac{\partial^2}{\partial k^2}\Big]h_{\mathrm{u}}(u,k) = \frac{1}{2}T'(u)\vartheta(k)^2 h_{\mathrm{kk}}(u,k) \tag{4.14}$$

Usando il fatto che (4.13) preserva la convessità abbiamo che se $g(0,\cdot)$ è convesso allora

$$h_{\mathrm{u}}(u,k) \geq 0 \quad \forall (u,k). \tag{4.15}$$

Si conclude quindi che i prezzi delle opzioni generati da (4.1) e (4.5) sono consistenti con l'assenza di arbitraggio.

□

4.3 Implementazione

In questa sezione vengono esposti gli algoritmi utilizzati per implementare il modello A-H e i relativi commenti ad essi. Il primo algoritmo riguarda la funzione MAIN, la struttura portante del modello, per poi passare alla spiegazione delle funzioni secondarie. Prima di ogni algoritmo di quest'ultime è presente una breve descrizione, così da poter essere consultata velocemente durante la lettura della parte riguardante la funzione MAIN.

Algoritmo 1: AH - Main Function

```
/* - parte 1 -                                                    */
Inizzializza le variabili e la memoria necessaria;
Dati: vettore K degli strike
Dati: vettore T degli istanti temporali (le diverse maturity presenti)
foreach Istante temporale→ i do
    if si trova nel primo istante then
        non considera l'incremento;
    else
        lunghezza del passo(i) ← T(i) − T(i − 1);
    end
end;
Dati: Matrice impVol formata dalle volatilità implicite prese dal
      mercato
foreach Istante temporale → i do
    foreach strike → k do
        Prezza un opzione Call ←
        B&S(S_0, T = i, strike = k, volat = impVol(i, k));
        Riempie la matrice R (chiamata prezzi di mercato);
    end
end;
/* - parte 2 -                                                    */
foreach Istante temporale→ i do
    salva nel vettore OO(i) il numero di valori ≠ 0 della colonna i di
    ImpVol;
end;
foreach strike do
    Calcola la condizione iniziale e la salva nel vettore C;
end;
Prima colonna della matrice dei prezzi prezziTOT ← C;
```

```
/* - parte 3 -                                                    */
for  i = seconda to ultimascadenza do
    Crea il vettore dei lungo come OO(i) per le volatilità (sigmazero);
    for j = primo to ultimostrike do
        if l'elemento della matrice ImpVol corrispondente a scadenza i
        e strike j è ≠ 0 then
            riempie il primo elemento vuoto del vettore appena creato
            con il valore di ImpVol(i, j);
        end
    end;
    PP ← prezzi reali R all'istante i;
    CC ← prezzi generati dal modello all'istante i − 1;
    /* - parte 3.1 -                                              */
    x ← runnested(CC,PP,sigmazero,impVol,istante i-esimo,...);
    sigmaVET ← AndreasenHuge(# di Strike, x, impVol, istante
    i-esimo);
    /* - parte 3.2 -                                              */
    C ← DiffFin(sigmaVET, passo temporale i-esimo, Strike, prezzi
    all'istante precedente);
    sigmaTOT(:, i) ← sigmaVET;
    riempie la colonna corrispondente con le volatilità appena calcolate;
    PrezziTOT(:, i) ← C;
    riempie la colonna corrispondente con i prezzi appena calcolati;
    /* - parte 3.3 -                                              */
    for j = 1 to 10 do
        calcola i prezzi per t_j ∈]t_i, t_{i+1}[ non presenti nella griglia iniziale
        utilizzando le formule (4.5) e (4.6);
    end;
end;
/* - parte 4 -                                                    */
for  i = prima to ultimascadenza do
    for j = primo to ultimostrike do
        if l'elemento della matrice impVol corrispondente a scadenza i
        e strike j è ≠ 0 then
            err ← somma errori (R(j, i) − prezziTOT(j, i)) al quadrato;
        end
    end
end;
err ← calcola l'errore quadratico medio;
```

Nella PARTE 1 del codice principale (Main) vengono inizializzate tutte le variabili utilizzate nel resto dell'algoritmo. Come **input** esterni al codice, vengono inseriti il vettore K contenente gli strike (lungo quanto il numero di righe della matrice delle volatilità) e il vettore T contenente le maturity (lungo quanto le colonne). Tramite *ciclo for* viene inizializzato anche il vettore delle differenze delle maturity, così da avere indicazione del passo temporale sempre diverso, come si è potuto osservare dalla Figura 4.1(a). In input viene passata anche la matrice delle volatilità implicite (A.1) presenti sul mercato, modificata come suggerito nell'Appendice A precedentemente in Excel e quindi salvata direttamente in $impVol$.

E' stata calcolata poi la superficie dei prezzi di mercato associati a tali volatilità. Per fare ciò viene utilizzata una funzione già presente in MATLAB che calcola il prezzo di un opzione Call[2] tramite Black&Scholes in forma chiusa dati lo strike, il tasso di interesse, il valore del sottostante all'istante iniziale S_0, la scadenza e la volatilità. In questo caso viene creata una matrice R dimensionalmente uguale a $ImpVol$ nella quale viene inserito nella posizione (i, j) il prezzo calcolato dalla funzione con strike uguale a $K(j)$, maturity $T(i)$, e volatilità uguale a $ImpVol(i, j)$, con tasso d'interesse e S_0 costanti per tutte le posizioni. Una volta costruita tutta la matrice si avrà la superficie dei **prezzi di mercato**. Si crea quindi una certa biunivocità tra i punti della superficie della volatilità implicita acquisita, e quelli della superficie dei prezzi generata utilizzando la implicita; poiché nella realtà sono i prezzi ad essere negoziati, e le volatilità vengono estratte invertendo la formula di Black&Scholes, si è semplicemente invertito il procedimento. Questo passaggio risulta utile, qui come negli altri modelli analizzati, per poter sempre passare da una superficie all'altra, da un elemento della matrice all'altro, in modo immediato e assecondando la necessita del momento. I prezzi generati servono tuttavia solo per testare l'accuratezza delle stime prodotte partendo da stesse volatilità, ma utilizzando algoritmi diversi, quindi l'unico requisito è che vengano mantenuti costanti per tutti i test fatti[3].

Una precisazione va fatta circa la forma della matrice delle volatilità in ingresso. Come già esposto nell'Appendice A riguardante i dati di mercato, la tabella scaricata è stata precedentemente modificata per rispondere alle esigenze di implementazione. $ImpVol$ non è nient'altro quindi che la matrice iniziale allargata, alla quale cioè sono state aggiunte righe intermedie

[2]Sarebbe stato del tutto equivalente se avessimo creato e impostato tutto il lavoro basandoci su opzioni Put.

[3]Senza escludere inoltre il fatto che, sebbene largamente dimostrata l'inesattezza della formula Black&Scholes, nella pratica è molto diffusa appunto per la sua velocità di utilizzo, per avere un prezzo *dirty* ma immediato.

per rendere la dimensione negli strike lineare (di passo 2,5%) e soprattutto ne sono state aggiunte sopra e sotto gli strike a disposizione per considerare valori molto lontani da quello at-the-money, cioè quello uguale al valore S_0. Questo passaggio è fondamentale per evitare che le condizioni al bordo (4.4) della matrice delle differenze finite, cioè gli estremi del dominio, particolarmente stringenti, siano troppo vicine alla zona di interesse e ne influenzino l'esito. La scelta di allargare la matrice aggiungendo strike fortemente ITM e OTM per gestire le condizioni al bordo e tra quelli presenti per renderli equispaziati è dipesa principalmente dal fatto che così facendo è possibile utilizzare la stessa funzione AndreasenHuge sia per i vettori corti *OO* formati solo dai valori non nulli presenti sia per i vettori lunghi tutta la griglia semplicemente cambiando i parametri in ingresso di volta in volta, rendendo il codice più comprensibile e all'occorrenza facilmente modificabile.

Nella PARTE 2 viene inizializzato il vettore *OO* lungo come il numero di maturity a disposizione, contenente il numero di volatilità diverse da zero presenti per ogni slice temporale e servirà per poter creare i vettori ogni volta di lunghezze diverse, necessari per l'ottimizzazione e per il confronto.

Il vettore C è il vettore dove vengono salvati i prezzi ottenuti dal modello ad ogni iterazione; viene inizializzato secondo la condizione iniziale presente nella formula di Dupire (4.1) e usato come vettore di partenza per le differenze finite; ad ogni iterazione quindi contiene i prezzi ai quali viene applicata la matrice per creare i prezzi dell'istante successivo.

La PARTE 3 è il punto cardine del modello di Andreasen e Huge, e viene sviluppata sorprendentemente in un unico *ciclo for*. Per ogni istante temporale vengono:

- Creati e riempiti i vettori lunghi come il numero di valori della volatilità diversi da zero in quell'istante;
- Risolto tramite la funzione Runnested il problema di ottimizzazione vincolata;
- Dato in input alla funzione AndreasenHuge il valore ottimo, la quale restituisce il vettore costante a tratti modificato delle volatilità, lungo quanto l'intera griglia;
- Calcolato il vettore dei prezzi all'istante di tempo successivo attraverso la funzione DiffFin;
- Riassemblati i vettori per poter essere memorizzati e utilizzati per l'istante successivo.

Nel dettaglio: nella prima parte del ciclo vengono creati i vettori che serviranno per l'ottimizzazione; poiché ad ogni slice temporale il numero di valori presenti in $ImpVol$ è differente, e dovendo utilizzare solamente quelli per la ricerca del minimo, si è dovuto scandire il vettore dell'istante i e salvare i soli valori positivi $ImpVol(i, j)$ nel vettore delle volatilità *sigmazero*. Sempre nell'ottimizzazione si è utilizzato invece l'intero vettore dei prezzi di mercato all'istante i e dei prezzi generati dal modello all'istante $i - 1$, sarà chiarito in seguito il motivo. In teoria il vettore corto delle volatilità potrebbe essere inizializzato in modo arbitrario, perché funge solo da guess iniziale per la ricerca dell'ottimo; in pratica però, la presenza di *minimi locali* potrebbe produrre parecchi problemi nella stima del vettore, si è quindi optato per fornire come dato di partenza il vettore stesso delle volatilità di mercato. Questo fa sì che il minimo sarà un vettore diverso (poiché si minimizzano le distanze tra i prezzi e non tra le volatilità) ma non si allontanerà troppo dalla guess iniziale, potendo così considerare attendibili le stime ottenute.

PARTE 3.1: I tre vettori ottenuti vengono dati in input alla funzione RUNNESTED la quale restituisce il vettore contenente le sole volatilità non nulle, le ϑ_{ik} in riferimento alla matrice (4.3). Questo vettore viene a sua volta dato in input alla funzione ANDREASENHUGE che ha lo scopo di "allungarlo" e/o renderlo costante a tratti tra un valore di volatilità scaricato e l'altro.

PARTE 3.2: Insieme al vettore dei prezzi all'istante temporale $i-1$, il vettore costante a tratti delle volatilità viene dato in ingresso alla funzione DIFFFIN che tramite differenze finite calcola tutto il vettore dei prezzi all'istante i: in pratica calcola i prezzi immaginando che la volatilità appena ottenuta rimanga costante anche nella dimensione del tempo, così che il passo del metodo delle differenze finite coincida con quello temporale $T(i)$ della griglia.

Nella parte finale del ciclo vengono salvati i vettori calcolati da ANDREASENHUGE e DIFFFIN in matrici dimensionalmente uguali alla $ImpVol$. Salvandovi i vettori di volta in volta alla fine del ciclo, si ha sia una traccia dell'evoluzione nel tempo di questi valori, sia una struttura adeguata per un confronto finale: queste matrici di volatilità *sigmaTOT* e dei prezzi *prezziTOT* rappresenteranno il corrispettivo, creato dal modello, delle superfici di volatilità e dei prezzi di mercato, con stesse dimensioni e stessi valori di strike e scadenze.

PARTE 3.3: Fa ancora parte del ciclo-for e fondamentalmente serve per risolvere il problema delle scadenze t comprese tra t_{i} e $t_{\text{i+1}}$ come in (4.5) e (4.6). Nel caso si volesse calcolare il prezzo per un solo t non presente nella griglia iniziale, questa parte potrebbe essere sostituita con un semplice richiamo alla funzione delle differenze finite dandogli in input lo specifico t

e con la theta uguale a quella del t_{i}. Si è optato per infittire la griglia di un certo numero di volte (in questo caso 10) così da iterare il processo per 10 punti compresi tra t_{i} e $t_{\mathrm{i+1}}$ $\forall i$, in ognuno dei quali si è tenuta costante la ϑ_{ik} del t_{i}. Così facendo si è lasciata una certa omogeneità con gli altri modelli presentati in questa tesi, dove per motivi differenti sono state usate diverse suddivisioni degli intervalli tra scadenze. Dividendo ogni scadenza iniziale con lo stesso numero di sotto intervalli equispaziati fa sì che si abbia un infittimento maggiore nelle brevi scadenze, dove infatti serve maggior precisione nel calcolo dei prezzi.

Nella PARTE 4 conclusiva del codice viene calcolato l'errore quadratico medio del modello, il RMSE (Root Mean Squared Error): è la misura della differenza tra i valori stimati di un modello $\hat{y}_{\mathrm{i}}$ e i valori realmente osservati y_{i} più frequentemente utilizzata; è definito come

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(\hat{y}_{\mathrm{i}} - y_{\mathrm{i}})^2}{n}} = \sqrt{\mathbb{E}\big[(\hat{y} - y)^2\big]} \tag{4.16}$$

dal quale è evidente il motivo del nome root mean squared error. E' preferito rispetto al più semplice Absolute Mean Error $\frac{1}{n}\sum_{i=1}^{n}|\hat{y}_{\mathrm{i}} - y_{\mathrm{i}}|$ perché dà un peso maggiore ai pochi errori molto grandi, e quindi nei modelli dove la loro presenza è più probabile (per costruzione o per il numero elevato di stime molto diverse tra loro) dà una stima dell'errore più accurata e restrittiva.

In riferimento però al modello analizzato, questo procedimento viene eseguito solo in corrispondenza dei prezzi di mercato *reali*, cioè quelli corrispondenti alle volatilità diverse da zero. Pur avendo ottenuto infatti una matrice dei prezzi molto più fitta e senza interruzioni, quello che interessa realmente è ottenere una superficie di volatilità che principalmente si raccordi il più possibile nei punti dove esistono i prezzi reali, ottenendo i rimanenti di conseguenza. Scandendo quindi la matrice, si prendono quelli corrispondenti a valori diversi da zero nella matrice $impVol$ e si calcola la differenza al quadrato dai corrispondenti prezzi calcolati dal modello (quelli con stesso strike e stessa scadenza), se ne fa la media dividendo per il numero di prezzi considerati, cioè la somma degli elementi del vettore OO. Indicando con i, j la posizione dei valori non nulli si ha

$$err = \sqrt{\frac{\sum(R_{\mathrm{j}} - prezzoTOT_{\mathrm{j}})^2}{n}}, \quad n = \sum_{i} OO(i)\ .$$

FUNZIONE RUNNESTED

La funzione Runnested risolve il problema dell'ottimizzazione vincolata per la determinazione delle volatilità ϑ ad ogni istante temporale.

Algoritmo 2: AH - Ottimizzazione Vincolata

Function [x, fval] = `runnested` (CC, PP, sigmazero$\leftarrow x_0$, passo temporale$\leftarrow$DT, Strike,impVol)

Viene definita la funzione obbiettivo;
[x, fval] = fmincon(@(x) NESTEDFUN(x,CC,PP,DT,K,impVol), x_0 ,[ ],[ ],[ ],[ ],[ ],[ ], @ CONFUN);

%Ricerca quel vettore x tale che minimizza la funzione obbiettivo NESTEDFUN, *partendo dal valore iniziale x_0 e soddisfacendo sempre i vincoli in* CONFUN;

end;

Function f = `Nestedfun`(x,CC,PP,DT, K,impVol)

sigmaVET $\leftarrow$ `AndreasenHuge(`*# di Strike, x, impVol, istante i-esimo*`)`;

DD $\leftarrow$ `DiffFin(`*sigmaVET, DT, Strike, CC*`)`;
prezzo generato da un passo delle differenze finite con volatilità= x;

for *j = primo* **to** *ultimostrike* **do**
  if *l'elemento della matrice impVol corrispondente allo strike j è* $\neq 0$ **then**
    err $\leftarrow$ somma errori $(PP(j) - DD(j))^2$;
  end
end;
$f \leftarrow$ err;
% calcola la distanza al quadrato di ogni prezzo generato DD da ogni prezzo reale PP con volatilità non nulla;

end;

Tramite la funzione FMINCON, già presente in MATLAB, si cerca quella x che minimizzi l'output f della funzione NESTEDFUN rispettando però i vincoli presenti in CONFUN.

La funzione NESTEDFUN quindi produce per ogni vettore di volatilità x i prezzi generati da un passo delle differenze finite tramite la funzione DIFFFIN, e poi ne calcola la distanza dai prezzi reali chiamati PP; restituendo questa distanza e minimizzandola si ottengono i ϑ che, qualora utilizzati nella matrice (4.3) per generare i prezzi, riproducono meglio quelli di mercato.

Dal momento che il vettore iniziale dell'ottimizzazione x_0 ha dimensione diversa ad ogni iterazione (è lungo quanto $OO(i)$) è stato necessario inserire nella funzione obbiettivo NESTEDFUN anche la funzione ANDREASENHUGE che estende questo vettore corto a tutta la griglia mantenendolo costante a tratti. Questo passaggio risulta essenziale perché la ricerca dei *pochi* ϑ in ogni instante temporale deve tener conto del fatto che poi questi valori verranno stirati su tutta la griglia e serviranno a generare i prezzi successivi; per assurdo infatti se esistessero tutte le volatilità per ogni strike, si potrebbe ottimizzare senza la funzione ANDREASENHUGE all'interno, ma chiamandola successivamente con il risultato ottenuto, poiché si otterrebbe il medesimo risultato non dovendo stirare il vettore e lasciando libero ogni ϑ di assumere valori molto diversi tra loro.

Per calcolare la distanza tra i prezzi non è stato possibile utilizzare la norma 2, detta anche norma euclidea[4], applicata al vettore formato dalla differenza tra i due vettori dei prezzi poiché il confronto va fatto solo sui prezzi associati a volatilità non nulle. E' stato quindi necessario scandire il vettore e calcolare la somma delle distanze al quadrato in quei soli punti, e poi minimizzare tale valore.

VINCOLI

La funzione confun contiene i vincoli utilizzati nella funzione Runnested.

Algoritmo 3: AH - Vincoli

```
Function [C,Ceq] = confun (parametri da vincolare)
    foreach elemento λ del vettore dei parametri do
        C ← [ λ > 0 ; λ < 100% ];
        Ceq ← [ ] % nessuna condizione di uguaglianza;
    end;
```

Questi vincoli impongono che ogni elemento da minimizzare (nel nostro caso ogni elemento del vettore delle volatilità) assuma valori compresi tra 0% e 100%. Per costruzione si è dovuta aggiungere anche una condizione di uguaglianza, ma nel modello non è stato necessario specificarla.

[4]Prendendo un vettore generico $X = (x_1, x_2, \ldots, x_n)$ la sua norma 2 nello spazio euclideo n-dimensionale $\mathbb{R}^n$ diventa:

$$\|X\| := \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}\ . \tag{4.17}$$

FUNZIONE ANDREASENHUGE

Sia che venga utilizzata nell'ottimizzazione per ricercare il minimo, sia nella funzione main per generare le volatilità locali vere e proprie, questa funzione è sempre utilizzata per rendere un vettore di volatilità costante a tratti.

Algoritmo 4: AH - costante a tratti

```
Function [piecewise] = AndreasenHuge (# di strike, sigma, impVol,
istante i-esimo)
    punti ← posizione delle vola maggiori di 0 nella matrice impVol;
    % Inizializza e riempie i vettori della volatilità e degli intervalli;
    for k = primo to ultimopunto do
        interv(k) ← intero più vicino al valore medio tra due punti con
        volatilità maggiori di 0;
    end;
    Inizia considerando il primo elemento del vettore delle volatilità
    stimate sigma (k=1);
    for j = primo to ultimostrike do
        % riempie il vettore in modo che sia costante a tratti;
        if non si è ancora alla fine dell'intervallo k-esimo then
            piecewise(j) ← elemento k-esimo di sigma;
        else
            if non si è ancora all'ultimo punto then
                Considera l'intervallo successivo;
                k = k + 1;
            end;
            piecewise(j) ← elemento k-esimo di sigma;
        end;
        end;
end;
```

Questa funzione prende in ingresso il numero di strike totali, il vettore delle ϑ calcolato con l'ottimizzazione vincolata e denominato *sigma*, la matrice delle volatilità di mercato $impVol$ e l'istante i-esimo di tempo considerato.

Prima di passare alla spiegazione dell'implementazione della funzione, bisogna fare un passo indietro per esporre una caratteristica della matrice (4.3). La funzione volatilità locale $\sigma^2_{\mathrm{LV}}(K,t)$ presente nell'equazione di Dupire (2.2) può essere sostituita da n approssimazioni nel tempo $\vartheta_{\mathrm{i}}(k)$, formate da fun-

zioni costanti a tratti negli strike:

$$\vartheta_{\mathrm{i}}(k) = \begin{cases} a_1 & K \leq b_1 \\ a_{\mathrm{ij}} & b_{\mathrm{ij}-1} < K \leq b_{\mathrm{ij}} \quad ij = 2, \ldots, m_{\mathrm{i}} - 1 \\ a_{\mathrm{m_i}} & b_{\mathrm{m_i}-1} < K \end{cases}$$

dove i b_{ij} sono i punti medi degli strike corrispondenti ai valori di mercato di un'assegnata maturity t_{i}:

$$b_{\mathrm{ij}} = \frac{K_{\mathrm{ij}+1} - K_{\mathrm{ij}}}{2} \qquad ij = 1, \ldots, m_{\mathrm{i}} - 1. \tag{4.18}$$

così l'approssimazione i-esima della volatilità è determinata da m_{i} livelli di volatilità, uguali al numero di livelli di strike osservati sul mercato alla scadenza t_{i}.

Per poter implementare questa suddivisione in livelli costanti, la funzione cerca e salva nel vettore *punti* gli strike corrispondenti sempre e solo a valori di volatilità maggiori di zero della matrice reale $impVol$ nell'istante i-esimo, per poi calcolare i punti intermedi, cioè gli estremi degli intervalli dove le volatilità rimarranno costanti, chiamati $interv$[5].

Il vettore costante a tratti viene creato scandendo gli elementi tante volte quanti il numero di strike totali, in modo tale da riempire interamente il vettore lungo quanto l'altezza della griglia. Nel ciclo-for si scorre tutto il vettore tenendo traccia dell'intervallo dove ci si trova, salvando nel vettore *piecewise* il valore assegnato a quell'intervallo prendendo i valori a_{ij} direttamente dalla posizione j del vettore *sigma* (in quanto è formato esattamente dagli strike dei valori $\neq 0$), fino a quando si passa all'intervallo successivo.

Un esempio di come lavora questa parte di codice è presente in Figura 4.3, dove per maturity di 1 mese e di 5 anni sono stati rappresentati sullo stesso grafico le corrispondenti stime x fatte dalla funzione RUNNESTED nei punti associati ai valori maggiori di zero in $impVol$ e il vettore di output $sigmaVET$ della funzione ANDREASENHUGE che rappresenta la volatilità locale. Guardando i grafici si possono fare diverse considerazioni: i livelli dove la volatilità resta costante cambiano (dove possibile) nel punto medio tra gli strike associati a due valori di mercato, in linea con la condizione (4.3).

[5]In riferimento alle formule appena esposte, *punti* corrisponde al vettore contenente i K e $interv$ a quello contenente i b_{ij}. Per questioni implementative si è dovuto considerare l'intero inferiore per ogni componente di $interv$, così da ottenere gli estremi degli intervalli in linea con la griglia a disposizione e quindi da poter essere utilizzati nel resto del codice senza problemi.

La volatilità a 5 anni ovviamente approssima meglio quella reale perché avendo più punti a disposizione gli intervalli sono mediamente più corti e coprono maggiormente la larghezza della superficie (si osservino i valori lontani dallo strike 100%, nel grafico 'un mese' già per valori sopra 110% e sotto il 70% viene considerato tutto costante, mentre nel 5 anni ci si può spingere fino a 160% e 30%). I livelli di volatilità locale ottenuti ricordano l'andamento della volatilità implicita di mercato, cioè hanno una pendenza più lieve per scadenze maggiori e le volatilità riferite a strike OTM sono generalmente più alte rispetto a quelle ITM; tuttavia in alcuni punti si ha maggiore irregolarità dovuta al fatto che i vari tasselli che compongono la volatilità locale in questo modello sono liberi di assumere qualsiasi valore, indipendentemente dai vicini, purché i prezzi associati approssimino il più possibile quelli di mercato.

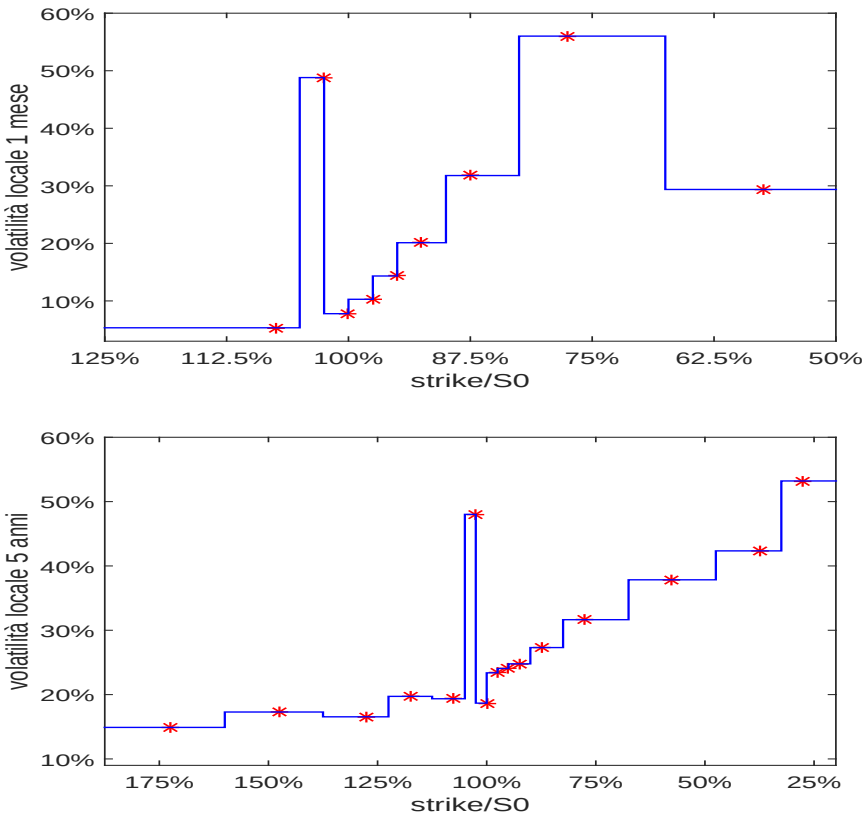


Figura 4.3: Volatilità stimate e funzione volatilità locale costante a tratti generata da AndreasenHuge per differenti maturity.

FUNZIONE DIFFFIN

La funzione DiffFin utilizza la stessa matrice (4.3) **sia per calcolare i prezzi necessari alla funzione Runnested per il confronto con quelli reali e la determinazione del vettore ϑ migliore, sia per calcolare effettivamente tutti i prezzi della slice temporale considerata;**

Algoritmo 5: AH - Differenze Finite

Function [Prezzo] = `DiffFin` (sigma, passo temporale, Strike, u←prezzo al tempo t-1)

%Inizializza le variabili necessarie per risolvere un passo delle differenze finite;
dtau ← passo temporale; dK ← 2.5%;
g ← $0.5 * (dtau/dK^2) * sigma^2 * K^2$;
% Crea una matrice A di dimensione $N \times N$ con N=101;
A ← matrice tridiagonale [-g ; 1+2*g ; -g];
u ← A/u ;
% applica la matrice A al vettore dei prezzi u;
Prezzo ← u;

end;

La matrice tridiagonale ha sempre la stessa dimensione, cioè 101×101, ovvero quanto il numero totale di strike considerati come spiegato nell'Appendice A, infatti non dipende dalla parte di codice dove la funzione DIFFFIN è impiegata: Questo grazie alla funzione ANDREASENHUGE che indipendentemente da cosa riceve in ingresso, restituisce sempre un vettore lungo 101.

Nel caso dell'utilizzo nell'ottimizzazione, la funzione DIFFFIN ha lo scopo di creare i prezzi C_{i+1} (secondo la notazione presente nella spiegazione del *core del modello*, partendo dai prezzi C_i) così da poter essere confrontati coi prezzi di mercato $\widetilde{C}_{i+1}$, basandosi sulle volatilità che si stanno ottimizzando. Nel secondo caso invece utilizzando l'ultima versione del vettore delle volatilità[6] la funzione dà in output i prezzi veri e propri del modello, che andranno a riempire la slice corrispondente alla determinata scadenza e formeranno la superficie dei prezzi. Più che la superficie comunque lo scopo principale di questi prezzi è quello di fornire il vettore sul quale basarsi per stimare

[6]cioè quello che verrà aggiunto esattamente com'è nella matrice che alla fine formerà la superficie di volatilità *sigmaTOT*.

la superficie di volatilità locale e in seguito creare i prezzi della scadenza successiva, iterando poi il meccanismo.

4.4 Risultati

Lo scopo principale del modello è quello di creare una superficie locale completa data la superficie di volatilità implicita scaricata da mercato. Una volta quindi calcolata e generati i prezzi utilizzando il metodo delle differenze finite, sono stati poi confrontati con quelli di mercato per vedere qual è l'errore che si può commettere utilizzando il modello al posto dei dati di mercato, e quanto la superficie generata interpoli bene quest'ultimi. Si è tenuta traccia inoltre del tempo di esecuzione del codice, in quanto nel mondo dell'ingegneria finanziaria (ma vale anche in tutta l'ingegneria) è un indice di primaria importanza che può determinare la preferenza di un metodo piuttosto che un'altro.

Nella Figura 4.4 è rappresentata la superficie di volatilità con i soli dati di mercato ma già equispaziata negli strike: rappresentando direttamente la matrice in Tabella A.1 infatti si avrebbe dato più l'idea della **superficie**, ma meno della vera pendenza e grandezza. Partendo da questa come input iniziale ed ottimizzando le volatilità ϑ, il modello di Andreasen-Huge produce la superficie di volatilità locale presente in Figura 4.5.

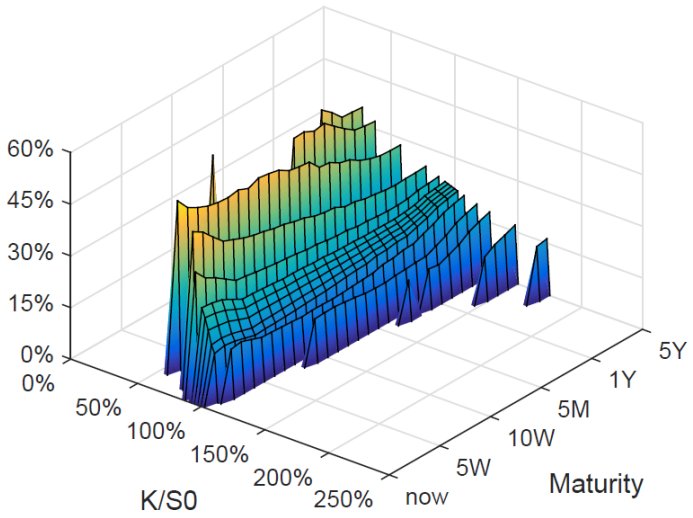


Figura 4.4: Superficie della volatilità implicita con i soli valori dal mercato.

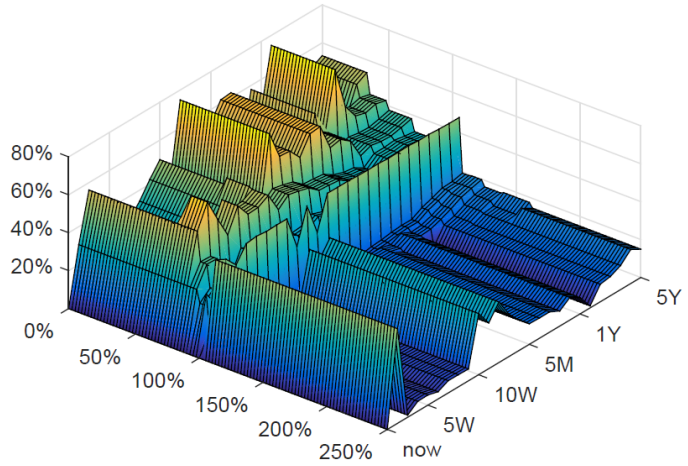


Figura 4.5: Superficie di volatilità locale generata dal modello A-H, al variare del tempo e di K/S_0.

Salta subito all'occhio che per valori lontani da quelli inizialmente presenti nella superficie implicita in Figura 4.4, la superficie si allarga in modo *flat* e a meno di qualche irregolarità si notano volatilità maggiori in corrispondenza di strike minori, andando via via a decrescere come già accennato nella Figura 1.1(b) a proposito del mercato equity. Un'altra proprietà che ci si aspettava, e osservando la figura difatti si può notare è la pendenza della superficie: molto alta per brevi scadenze, per poi appiattirsi all'allungarsi del tempo di vita dell'opzione.

Utilizzando la superficie locale in Figura 4.5 è stato possibile calcolare i prezzi per ogni strike e per ogni maturity presenti nella griglia considerata tramite la formula discretizzata di Dupire (4.1). In Figura 4.7 è possibile osservare questi prezzi al variare dei due parametri K e T.

Viene riportato in Figura 4.6 il grafico dei prezzi calcolati con Black&Scholes in forma chiusa utilizzando le sole volatilità della Tabella A.1, con le stesse maturity e strike, così da lavorare con le stesse dimensioni e riferimenti. Il grafico dei prezzi è stato fatto volutamente a pallini, non come superficie, per rendere maggiormente l'idea che si ha a disposizione solo un numero discreto di valori dal mercato, giustificando la necessità di dover riempire tutta la griglia (evitando tuttavia la possibilità di arbitraggio) per poter avere a disposizione ogni prezzo ed utilizzare le formule della volatilità locale.

Per capire quanto questo metodo produca una superficie utilizzabile e che interpoli il più possibile i dati di mercato, in Figura 4.7 è stato fatto

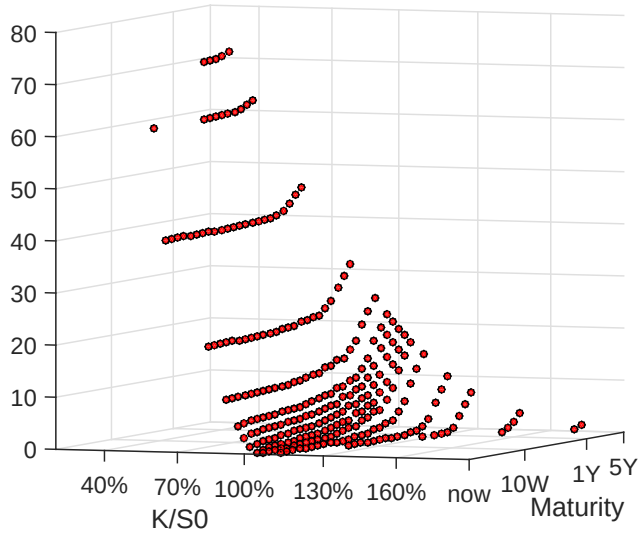


Figura 4.6: Prezzi dell'opzione Call con volatilità non nulle.

uno zoom della parte centrale del grafico e sono stati sovrapposti i prezzi di mercato (in Figura 4.6) ai prezzi generati dal codice. Si può notare che quasi tutti i pallini rossi vengono raggiunti dalla superficie senza buchi o salti di quest'ultima, anche quelli molto lontani dall'ATM e con scadenze ampie; i prezzi generati sono crescenti sia nella variabile tempo che nella variabile strike, garantendo quindi l'assenza di arbitraggio.

modello	Root Mean Squared Error	Leave-one-out	tempo
A-H	0.0022	0.0198	20.651 s

Tabella 4.1: Errore e tempo di esecuzione del modello Andreasen-Huge.

L'errore complessivo del modello, presente in Tabella 4.1, è molto piccolo, soprattutto se si considera il fatto che nella pratica un prezzo può ritenersi veritiero, utilizzabile, se dista dal prezzo di mercato al massimo 10 basis points; questa misura è l'ampiezza media del **bid-ask spread**, cioè la differenza tra il prezzo al quale il mercato è disposto a vendere un determinato prodotto (bid) e il prezzo al quale è disposto ad acquistarlo (ask)[7]. Alla luce di ciò possiamo ritenere il modello di Andreasen e Huge in linea con le richieste di precisione dettate dai trader e practitioners.

[7]Le volatilità scaricate da Bloomberg e utilizzate in questa tesi si riferiscono al solo prezzo *mid*, cioè la media tra i *bid* e *ask* e rappresenta il prezzo equo, senza influenze di breve termine del mercato, del prodotto in questione.

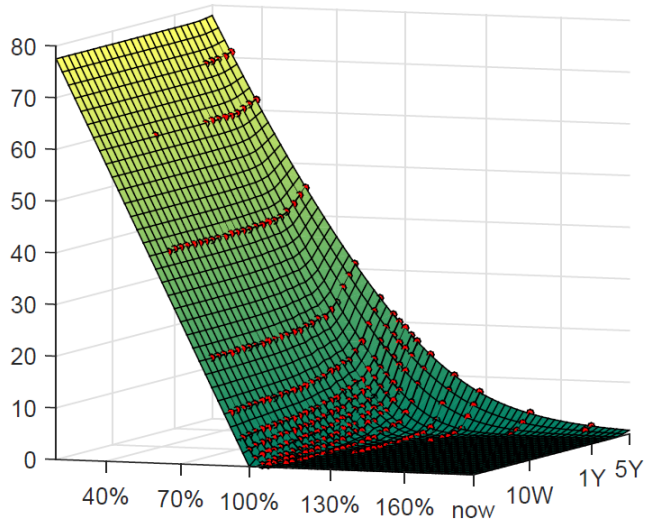


Figura 4.7: prezzi di mercato (pallini rossi) e prezzi ottenuti dal modello A-H, al variare del tempo e di K/S_0.

Il Leave-one-out è un metodo per valutare la bontà della calibrazione, e quindi del modello, quando si utilizzano dei dati sia per calibrare che per valutare l'efficacia finale: in pratica si esclude dal set iniziale un certo valore, si calibra tutta la superficie senza di esso e poi si va a controllare quanto dista il valore generato dal modello da quello originalmente presente. In questo caso si è scelto di escludere il prezzo della Call europea con scadenza 5 anni e strike 102.5% poichè è nella zona affetta da errori più grandi, e di calcolare il prezzo della stessa opzione tramite il metodo A-H; il valore presente in Tabella 4.1 non è altro che la differenza tra il prezzo originario e quello calcolato.

Per quanto riguarda il tempo, va specificato che varia a seconda del computer sul quale viene fatto girare il codice: nel caso in esame è stato usato un processore Intel Core i5-337U a 1.80GHz con 6GB di RAM.

4.5 Le modifiche proposte da Abasto, Hientzsch e Kust

Il secondo modello sviluppato in questa tesi per ottenere la superficie di volatilità locale in assenza di arbitraggio è stato proposto da Damian Abasto, Bernhard Hientzsch e Mark Kust [1] (indicato anche come A-H-K) come possibile modifica, ovvero come *step successivo*, al lavoro fatto da Andreasen e

Huge [4]. L'articolo dei tre autori viene proprio stilato in relazione al [4], elencandone i punti in comune e soprattutto le differenze introdotte per cercare di migliorare l'efficienza della calibrazione e dei risultati ottenuti. Queste modifiche hanno interessato sia l'orizzonte temporale della griglia, sia il metodo col quale sono stati creati i prezzi, sia l'ottimizzazione necessaria per ottenerli.

Anche in questo caso verrà innanzitutto esposto il procedimento e le variazioni in questione, per poi passare all'implementazione e alle spiegazioni delle scelte fatte, con i risultati ottenuti in coda al capitolo.

4.5.1 Dove agiscono queste modifiche

Basandosi per buona parte sul lavoro di Andreasen e Huge, anche questo modello si propone di risolvere l'equazione in avanti di Dupire

$$\frac{\partial C(T,K)}{\partial T} = \frac{1}{2}\sigma_{\mathrm{LV}}(T,K)^2 K^2 \frac{\partial^2 C(T,K)}{\partial K^2} \tag{4.19}$$

dato un insieme di scadenze $0 = t_0 < t_1 < \cdots < t_{\mathrm{n}}$ e tramite differenze finite implicite

$$\begin{gathered}\Big[1 - \frac{1}{2}\triangle t_{\mathrm{i}} \vartheta_{\mathrm{i}}(K)^2 K^2 \frac{\partial^2}{\partial K^2}\Big] c(t_{\mathrm{i+1}}, K) = \; c(t_{\mathrm{i}}, K), \\ c(0,K) = (S(t_0) - K)^+.\end{gathered} \tag{4.20}$$

$\vartheta_{\mathrm{i}}(K)$ rappresenta una "piastrella" della volatilità locale che ha la forma parametrica

$$\begin{gathered}\vartheta(t,K) = \vartheta_{\mathrm{ij}}, \quad t_{\mathrm{i-1}} < t < t_{\mathrm{i}} \\ \frac{(K_{\mathrm{j-1}} + K_{\mathrm{j}})}{2} < K \leq \frac{(K_{\mathrm{j}} + K_{\mathrm{j+1}})}{2}, \quad 1 \leq i \leq n, \quad 0 \leq j \leq J_{\mathrm{i}},\end{gathered} \tag{4.21}$$

in questo modo la funzione volatilità locale $\vartheta(t,K)$ è assunta essere costante a tratti nel tempo e nello strike, con tanti parametri da calibrare quanti valori di mercato disponibili alla scadenza t_{i}.

Anche in questo caso la differenza tra i prezzi di mercato e quelli del modello ottenuto risolvendo (4.20) e (4.21) è minimizzata attraverso l'ottimizzazione, ma in questo caso la funzione da minimizzare è diversa: in [4] Andreasen e Huge minimizzano esclusivamente la funzione $\vartheta(K)$, mentre Abasto, Hientzsch e Kust propongono di minimizzare direttamente il prodotto $K\vartheta(K)$, così facendo si restringe notevolmente lo spazio di ricerca dell'ottimizzatore utilizzato nella calibrazione.

La differenza più importante con il modello A-H riguarda però la **discretizzazione del tempo**. Nello schema di discretizzazione (4.21) la grandezza degli step $\triangle t_{\mathrm{i}}$ è impostata per essere il tempo tra due scadenze successive

nei dati di mercato scaricati. Sebbene questo renda più veloce la calibrazione (poiché viene risolto un solo una sistema tridiagonale per iterazione), rappresenta un'approssimazione troppo *cruda* della derivata rispetto al tempo $\partial/\partial T$.

Per evitare il problema, Lipton e Sepp [33] propongono di risolvere l'equazione di Dupire (4.19) in forma esatta attraverso trasformazioni di Laplace e funzioni di Green. La volatilità sarebbe ancora della forma (4.21) e sarebbe ottenuta tramite ottimizzazione vincolata, ma il metodo proposto incorpora un costo computazionale elevatissimo per le trasformate dirette e inverse di Laplace e gli integrali. La trasformata inversa di Laplace inoltre è rigida e può essere instabile, soprattutto per le scadenze brevi.

Al posto di produrre i prezzi di mercato (e quindi anche l'ottimizzazione) attraverso l'equazione (4.20), in [1] i tre autori propongono invece di infittire la griglia prendendo ulteriori step tra quelli originariamente presenti $\{t_{\mathrm{i}}\}_{\mathrm{i}=0}^{n}$, calibrando comunque i ϑ_{ij} sull'intero intervallo. Questo si tramuta nell'equazione

$$\begin{gathered}\Big[1-\frac{1}{2}\frac{\triangle t_{\mathrm{i}}}{N_{\mathrm{t}}}\vartheta_{\mathrm{i}}(K)^2K^2\frac{\partial^2}{\partial K^2}\Big]c(t_{\mathrm{i}}+\frac{\triangle t_{\mathrm{i}}}{N_{\mathrm{t}}},K)=c(t_{\mathrm{i}},K)\\ c(0,K)=(S(t_0)-K)^+,\quad i=0,1,\ldots\end{gathered}\tag{4.22}$$

dove N_{t} rappresenta il numero di sotto-intervalli tra le scadenze di mercato. Dal punto di vista numerico l'utilizzo di step temporali $\triangle t$ più piccoli si tramuta in un errore, che nel caso delle Differenze Finite è $O(\triangle x^2+\triangle t)$, più piccolo, e in un'approssimazione della derivata $\partial/\partial T$ migliore.

Il problema di ottimizzazione resta sempre lo stesso: trovare ϑ_{i} tali che minimizzino la distanza tra i prezzi di mercato e quelli generati dal modello alla scadenza t_{i}, ma in questo caso quest'ultimi prezzi vengono trovati dopo N_{t}-volte che si itera il processo pur mantenendo costante il valore delle volatilità in ogni sotto intervallo. Si osservi la Figura 4.8 nella quale è presente uno schema di come questo processo evolve, e lo si rapporti alla Figura 4.2 del processo di Andreasen-Huge: in A-H prima si calibrano le ϑ in riferimento all'intervallo più ampio e poi si utilizzano gli stessi valori per i sotto intervalli, qua invece si calibrano in modo tale che, se utilizzate con lo stesso valore ma in ogni intervallino e procedendo nel tempo si ottengano alla fine dell'intervallo iniziale t_{i} i prezzi il più vicino possibile ai prezzi di mercato con medesima scadenza.

Utilizzando il procedimento (4.22) inoltre è possibile ottenere fin da subito una griglia più fitta, e quindi un numero maggiore di prezzi, senza dover

riprezzare dall'istante iniziale qualora il prezzo che stiamo cercando abbia scadenza che stia su in sotto intervallo creato[8].

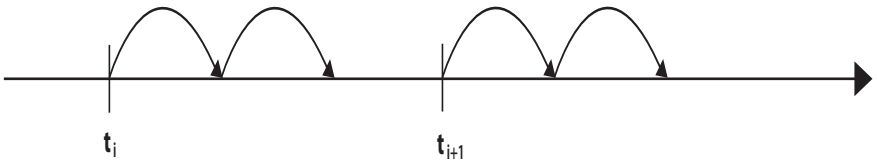


Figura 4.8: schema di avanzamento nel tempo del modello A-H-K.

Una volta calibrata e ottenuta tutta la superficie di volatilità locale, i tre autori propongono di calcolare la superficie dei prezzi attraverso il metodo Monte Carlo, spiegato nell'Appendice B, piuttosto che utilizzare nuovamente le Differenze Finite. Il motivo di questa scelta risiede nel fatto che la volatilità locale, per come è stata concepita, viene impiegata per calcolare i prezzi di derivati più complessi come le opzioni esotiche; poichè per tali opzioni non esiste ad oggi un'EDP specifica per ottenere il prezzo, non è possibile utilizzare metodi quali Differenze Finite o Elementi Finiti, dovendosi così rifare necessariamente alle simulazioni del metodo Monte Carlo. Abasto, Hientzsch e Kust quindi ragionando nell'ottica di dover prezzare strumenti più complessi delle banali europee indicano delle modifiche al modello tali da renderlo performante se associato al metodo Monte Carlo.

Il problema primario nell'utilizzo del Monte Carlo tuttavia risiede nel fatto che bisogna ipotizzare un possibile andamento del sottostante per poter generare le numerose simulazioni delle quali calcolarne poi la media. Ricordando però l'espressione differenziale (2.1) dell'andamento del sottostante ipotizzato nel lavoro di Dupire, si può notare la forte somiglianza col moto browniano geometrico (GBM)

$$dS_{\mathrm{t}} = rS_{\mathrm{t}}dt + \sigma S_{\mathrm{t}}dW_{\mathrm{t}}$$

del quale si conosce la forma esatta della discretizzazione dell'andamento

$$S(t_{\mathrm{i}}) = S(t_{\mathrm{i}-1})e^{(r-\frac{\sigma^2}{2})\triangle t+\sigma\sqrt{\triangle t}Z} \tag{4.23}$$

dove Z è una variabile aleatoria estratta dalla distribuzione normale standard. La variabile Z è l'unica fonte di casualità presente e deriva dalla

[8]Qualora si presentasse l'esigenza quindi, è possibile scegliere a priori gli intervallini, in modo tale da far combaciare una scadenza con quella di nostro interesse

proprietà del GBM che afferma che un processo di Wiener W_t ha incrementi stazionari e indipendenti e distribuiti come una normale di media zero e di varianza uguale all'ampiezza dell'intervallo:

$$\triangle W_t = W_{t+\triangle t} - W_t \sim N(0, \triangle t) = \sqrt{\triangle t} N(0,1).$$

Nelle formule dei moti browniani geometrici si deve assumere la volatilità costante per ogni strike e scadenza, rendendole all'apparenza non adatte ad un loro utilizzo in questo scenario. Tuttavia il fatto che nel modello di Dupire la volatilità sia ipotizzata costante a tratti, soprattutto nel tempo, ci dà la possibilità di utilizzare localmente queste formule, poiché localmente la volatilità è costante e quindi il processo evolve come un GBM tra una scadenza e l'altra. Per quanto riguarda invece il fatto che la superficie presenta una skew, quindi non è costante negli strike, la questione è un po' più delicata: considerando una singola simulazione per semplicità, bisogna controllare ad ogni passo della discretizzazione $S(t_i)$ quale valore assume il sottostante per poter cercare il livello di volatilità locale corrispondente e poterla utilizzare quindi nella formula (4.23) per ottenere $S(t_{i+1})$; il tutto va fatto simultaneamente per ogni simulazione, così che in ogni istante di tempo sia possibile calcolare il prezzo dell'opzione associata, oppure usarlo come nuovo punto di partenza per continuare l'evoluzione del sottostante nel tempo.

Una volta ottenute tutte le simulazioni fino a scadenza, è possibile calcolare i prezzi delle opzioni, in questo caso Call europee, per ogni strike e scadenza della griglia iniziale in modo immediato e confrontarle così coi prezzi di mercato per studiare la bontà del modello e delle modifiche proposte.

Assenza di arbitraggio

Nell'articolo di Abasto-Hientzsch-Kust [1] non viene menzionata la richiesta di assenza di arbitraggio; questo deriva dal fatto che basandosi su un modello nel quale ne è già stata garantita l'assenza, anche il modello qui esposto ne eredita le proprietà.

4.5.2 Implementazione

In questa sezione vengono esposti gli algoritmi utilizzati per implementare il modello A-H-K e i relativi commenti ad essi. Il primo algoritmo riguarda la funzione MAIN, per poi passare alla spiegazione delle funzioni secondarie. Prima di ogni nuovo algoritmo è presente una breve descrizione, così da poter essere consultata velocemente durante la lettura della parte riguardante la funzione MAIN.

Algoritmo 6: AHK - Main Function

```
/* - parte 1 -                                                    */
Inizzializza le variabili e la memoria necessaria;
Dati: vettore K degli strike
Dati: vettore T degli istanti temporali (le diverse maturity dei
      prodotti)
foreach Istante temporale→ i do
    if si trova nel primo istante then
        non considera l'incremento;
    else
        lunghezza del passo(i) ← T(i) − T(i − 1);
    end
end;
Dati: Matrice impVol formata dalle volatilità implicite prese dal
      mercato
foreach Istante temporale → i do
    foreach strike → k do
        Prezza un opzione Call ←
        B&S(S_0, T = i, strike = k, volat = impVol(i, k));
        Riempie la matrice R (chiamata prezzi di mercato);
    end
end;
/* - parte 2 -                                                    */
foreach Istante temporale→ i do
    salva nel vettore OO(i) il # di valori ≠ 0 della colonna i di ImpVol;
end;
foreach strike do
    Calcola la condizione iniziale e la salva nel vettore C;
end;
Prima colonna della matrice dei prezzi prezziTOT ← C;
```

```
/* - parte 3 -                                                    */
numero steps ← 10;
for i = seconda to ultimascadenza do
    Crea il vettore dei lungo come OO(i) per le volatilità (sigmazero) e
    per gli strike (KK);
    for j = primo to ultimostrike do
        if l'elemento della matrice ImpVol corrispondente a scadenza i
        e strike j è ≠ 0 then
            riempie il primo elemento vuoto dei vettori appena creati
            rispettivamente con il valore di ImpVol(i, j) e K(j);
        end
    end;
    PP ← prezzi reali R all'istante i;
    CC ← prezzi generati dal modello all'istante i − 1;
    sigmaKzero ← sigmazero×KK;
    x ← runnested(CC,PP,sigmaKzero,impVol,passo temporale
    i-esimo,...);
    x ← x/KK;
    sigmaVET ← AndreasenHuge(# di Strike, x, impVol, istante
    i-esimo);
    sigmaTOT(:, i) ← sigmaVET;
    riempie la colonna corrispondente all'istante temporale con le
    volatilità appena calcolate;
    sigmaK ← sigmaVET×KK;
    C ← DiffFin(sigmaK, passo temporale i-esimo, prezzi all'istante
    precedente, numero steps);
    alla prima iterazione riceve in ingresso la condizione iniziale, poi
    viene considerata come condizione iniziale il vettore dei prezzi al
    tempo t-1;
end;
/* - parte 4 -                                                    */
numero steps ← 100;
numero simul ← 50'000;
Z ← matrice contenente numeri casuali estratti da normale standard
N(0, 1);
[S,Sant] ← simGBM(prezzo iniziale, sigmaTOT, numero steps, numero
simul, ...);
```

```
for i = seconda to ultimascadenza do
    ST ← prezzo delle simulazioni S all'istante i-esimo;
    STant ← prezzo delle simulazioni Sant all'istante i-esimo;
    for z = primo to ultimostrike do
        calcola il prezzo al variare dello strike K;
        payoff ← max(ST − K(z), 0)e^{−rT(i)};
        payoffant ← max(STant − K(z), 0)e^{−rT(i)};
        media ← (payoff + payoffant)/2;
        C(z) ← stima del valore atteso della distribuzione normale
        dagli elementi di media;
    end;
    PrezziTOT(:, i) ← C;
    riempie la colonna corrispondente all'istante temporale con i prezzi
    appena calcolati;
end;
/* - parte 5 -                                                    */
for i = prima to ultimascadenza do
    for j = primo to ultimostrike do
        if l'elemento della matrice impVol corrispondente a scadenza i
        e strike j è ≠ 0 then
            err ← somma errori (R(j, i) − prezziTOT(j, i)) al quadrato;
        end
    end
end;
err ← calcola l'errore quadratico medio;
```

La prima differenza rispetto al metodo A-H riguarda la PARTE 3 del codice: viene inizializzato in questa parte di codice la variabile $Nsteps$, cioè il numero di sottointervalli (N_t in riferimento all'equazione (4.22)) nel quale suddividere il passo temporale originario. Si è optato per una divisione in dieci parti per ogni scadenza notando che già con 10 sotto-intervalli la stima era buona e soprattutto il costo computazionale notevolmente minore rispetto a griglie con maggior N_t.

Anche il modello proposto da Abasto, Hientzsch e Kust si basa sulla generazione della superficie locale tramite ottimizzazione in un unico *ciclo for*, molto simile a quello implementato da A-H. L'utilizzo del vettore KK è dovuto all'esigenza di lavorare in tutto il ciclo-for col prodotto $\vartheta(K)\cdot K$ per poterne trovare l'ottimo come suggerito dagli autori; quindi durante tutta la PARTE 3 si è dovuto moltiplicare o dividere il vettore delle volatilità per gli strike a

seconda del loro utilizzo. La funzione AndreasenHuge utilizzata in questo capitolo è esattamente quella implementata nel modello Andreasen-Huge, si rimanda quindi all'implementazione precedente per chiarimenti. Il vettore ottenuto da AndreasenHuge viene salvato nella matrice *sigmaTOT* che andrà poi a rappresentare la superficie di volatilità locale. Non vengono tuttavia immagazzinati i prezzi generati, ma semplicemente rinominati C così da poter essere utilizzati all'istante successivo: questo deriva dal fatto che in questa parte di codice lo scopo è solo quello di generare la superficie locale, lasciando alla parte successiva la creazione dei prezzi associati.

Nella PARTE 4 viene implementato il modello Monte Carlo per verificare la bontà del pricing del modello. Per prima cosa vengono decisi il numero di step temporali tra le maturity già presenti, e il numero di simulazioni necessarie per ottenere un buona stima della media tramite la legge dei grandi numeri. Si è osservato che con cento sotto-intervalli e cinquantamila simulazioni si possono ottenere risultati già soddisfacenti, contando inoltre sul fatto che utilizzando la tecnica di riduzione della varianza spiegata in Appendice B, queste simulazioni agiscono come se fossero il doppio, cioè 100'000.

Sono stati creati fin da subito tutte le componenti casuali delle simulazioni tramite funzione interna di MATLAB RANDN, la quale restituisce valori in $\mathbb{R}$ secondo la distribuzione di una normale standard con media zero e varianza uno.

Una volta generate tutte le componenti aleatorie del modello, vengono passate alla funzione simGBM la quale restituisce una matrice dove ogni riga (50'000) rappresenta l'evoluzione nel tempo di una simulazione del sottostante e avrà numero di colonne quante maturity e relativi sotto-intervalli (26X100); la funzione restituisce anche la matrice contenente le simulazioni generate utilizzando la variabile antitetica, da cui il suffisso -ant.

Dalle simulazioni S e S_{ant} è possibile quindi calcolare il payoff ed il prezzo dell'opzione cercata. Sempre grazie alle proprietà del GBM è possibile "tagliare" le simulazioni alle diverse scadenze presenti nella griglia iniziale ed utilizzare i valori assunti dal sottostante come valore a scadenza $S(T_{\text{i}})$. Dal momento che la superficie delle volatilità, e dei prezzi R, varia sia nel tempo che negli strike, fissata la scadenza il payoff dell'opzione viene calcolato al variare dei K, così da ottenere un vettore che in posizione z-esima ha il payoff corrispondente allo strike z, a parità di esiti delle simulazioni.

$$\begin{aligned} P &= \max(S(T) - K(z), 0)e^{-rT(i)} \\ P_{\text{ant}} &= \max(S_{\text{ant}}(T) - K(z), 0)e^{-rT(i)} \end{aligned} \tag{4.24}$$

dove $\max(S(T) - K(z), 0)$ è il payoff di un opzione Call europea. Utilizzando l'equazione (B.6) si può calcolare il prezzo finale $C(z)$ dell'opzione calcolando la stima campionaria del valore atteso del vettore $media = (P + P_{\text{ant}})/2$.

FUNZIONE RUNNESTED

La funzione Runnested risolve il problema dell'ottimizzazione vincolata per la determinazione delle volatilità $\vartheta(K)K$ ad ogni istante temporale. Praticamente la funzione svolge lo stesso compito di quella implementata in precedenza in A-H, con la sola differenza che si ricerca l'ottimo agendo sul vettore volatilità×strike ($\vartheta(K)K$) e non esclusivamente sulle volatilità.

VINCOLI

La funzione confun contiene i vincoli utilizzati nella funzione Runnested. Questi impongono che ogni elemento da minimizzare (ϑK) abbia il solo vincolo di essere maggiore di 0%.

FUNZIONE DIFFFIN

La funzione DiffFin calcola i prezzi necessari alla funzione Runnested per il confronto con quelli reali e la determinazione del vettore $K\vartheta(K)$ migliore. Viene utilizzata la stessa matrice tridiagonale (4.3) ma con il passo temporale $\triangle t$ diviso per $Nsteps$ e solo per calcolare i prezzi necessari alla funzione RUNNESTED per la determinazione del vettore ϑ migliore. Una volta ottenuta la matrice, viene applicata al vettore dei prezzi non una ma $Nsteps$ volte per ottenere il vettore dei prezzi desiderato.

FUNZIONE SIMGBM

La funzione simGBM ha lo scopo di creare tutte le simulazioni dell'andamento futuro del sottostante sia per la versione principale sia per la sua antitetica, facendo evolvere il sottostante secondo l'equazione (2.1)**.**

La funzione crea sia la matrice S che S_{ant} ed entrambe hanno la prima colonna, il punto di partenza di ogni simulazione, il sottostante al tempo zero S_0.

Il ciclo-for lavora su una colonna per volta, agendo come lo scorrere del tempo, su tutti i sotto intervalli (indicati dal pedice j). Tuttavia il fatto di avere la volatilità costante tra una scadenza principale e l'altra[9] fa sì che

[9]Si indica scadenza principale quella presente tra le 26 della griglia iniziale, per differenziarla dalle 100 scadenze create successivamente tra ognuna delle 26 originiali.

si debba tenere traccia della posizione anche rispetto a queste (indicate con pedice i).

Algoritmo 7: AHK - Simulazione sottostante

```
Function [S,Sant] = simGBM(sigmaTOT, prezzo iniziale← S_0, numero
steps, numero simul, Z, ...);
    %Inizializza le variabili necessarie per salvare i percorsi di ogni
    simulazione;
    S ← matrice numerosimul × (numerosteps ∗ numeroscadenze);
    Sant ← matrice numerosimul × (numerosteps ∗ numeroscadenze);
    prima colonna di S ← S_0;
    prima colonna di Sant ← S_0;
    foreach steps * (numero scadenze) do
        i ← scadenza precedente all'istante considerato;
        dt ← ampiezza sotto-intervallo tra T(i) e T(i − 1);
        si ricerca la volatilità raggiunta dal passo precedente delle
        simulazione;
        V ← AbastoHientzschKust(sigmaTOT(:,i),sottostante
        precedente←S(:,j-1),strike);
        Vant ← AbastoHientzschKust(sigmaTOT(:,i),sottostante
        precedente←Sant(:,j-1),strike);
        foreach simulazione ←y do
            S(y,j) ← formula esatta (4.23) con volatilità V;
            Sant(y,j) ← formula esatta (4.23) con volatilità Vant;
        end;
    end;
end;
```

Indicando con dt il passo da utilizzare in ogni scadenza tra quelle principali[10], la prima cosa da fare è ricercare per ogni simulazione la volatilità corrispondente al prezzo raggiunto dal sottostante all'istante precedente. Questo passaggio è necessario perché, seppur rimanendo la volatilità costante nel tempo durante tutti i sotto intervalli tra scadenze principali, può succedere che in una simulazione un prezzo salga o scenda molto in poco tempo, andando a finire in una zona dove la volatilità (costante a tratti negli strike, a prescindere dalla scadenza) assuma valori diversi da quelli utilizzati all'istante precedente. La ricerca della volatilità corrispondente viene implementata dalla funzione AbastoHientzschKust spiegata in seguito. Poiché le

[10]si ricorda rimandando alla figura 4.1(a) che l'intervallo tra scadenze è sempre diverso quindi lo è anche quello da i rispettivi sotto intervalli.

simulazioni con variabile antitetica chiaramente assumono valori differenti nel tempo, si è dovuta tenere traccia anche delle volatilità raggiunte dalle simulazioni antitetiche S_{ant}.

Per ogni simulazione ora si calcola il prezzo all'istante successivo tramite

$$S(y,j) = S(y,j-1) \cdot exp\Big\{ \frac{(r - V(y)^2)}{2} dt + V(y)\sqrt{dt} Z(y,j-1) \Big\}$$

$$S_{\text{ant}}(y,j) = S_{\text{ant}}(y,j-1) \cdot exp\Big\{ \frac{(r - V_{\text{ant}}(y)^2)}{2} dt - V_{\text{ant}}(y)\sqrt{dt} Z(y,j-1) \Big\}$$

dove si può notare che vengono utilizzate volatilità diverse e soprattutto nelle antitetiche la variabile casuale Z è presa col segno opposto per ridurre la varianza generale; y indica la simulazione e j il sotto intervallo presi in esame.

FUNZIONE ABASTOHIENTZSCHKUST

Questa funzione ricerca, per ogni simulazione ed ad ogni istante di tempo t_{i}, il valore raggiunto dal sottostante e restituisce il valore della volatilità locale associata.

Algoritmo 8: AHK - ricerca volatilità

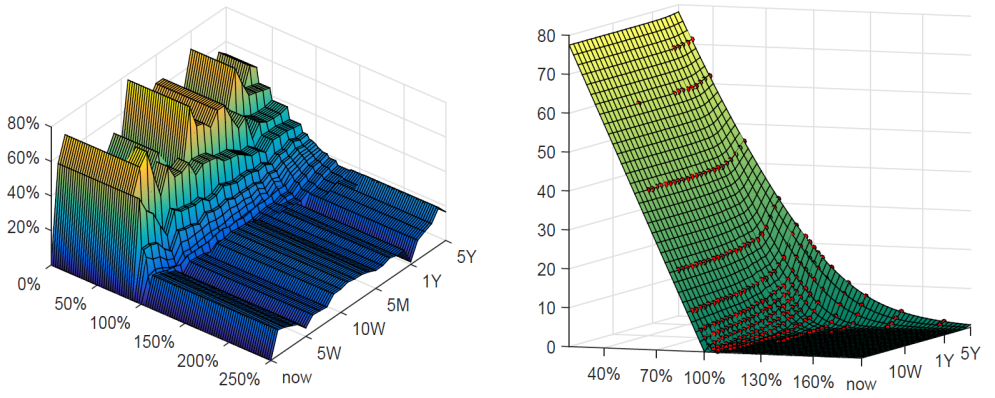
```
Function V = AbastoHientzschKust(sigmaVET, vettore dei
sottostanti←S, strike);
    %Inizializza il vettore delle volatilità raggiunte;
    for i = secondo to penultimostrike do
        ind ← funzione indicatrice di quali sottostanti sono arrivati tra
        punto medio(i-1,i) e punto medio(i,i+1);
        V(ind) ← sigmaVET(i);
        associa in ognuno di questi punti la volatilità corrispondente
        allo strike i;
    end;
end;
```

Riceve in ingresso il vettore contenente i prezzi raggiunti dalle simulazioni all'istante precedente S_{t}, gli strike K e il vettore contenente le volatilità locali stimate per l'ultima scadenza principale della griglia attraversata. Questo fa sì che durante tutti i sotto intervalli si abbiano valori diversi dei sottostanti S_{t} ma si prendano valori dalle stesse volatilità.

Per ogni valore dello strike, si ricerca nel vettore S_{t} la posizione di quelle simulazioni tali che all'istante considerato hanno prodotto un prezzo compreso tra $\frac{K(i-1)+K(i)}{2}$ e $\frac{K(i)+K(i+1)}{2}$ ed assegna a quelle posizioni il valore della volatilità associata allo strike $K(i)$.

4.5.3 Risultati

Partendo dai dati scaricati ed ottimizzando su $K\vartheta(K)$, utilizzando dieci passi temporali per ogni scadenza della griglia, il modello implementato produce la superficie di volatilità locale presente in Figura 4.9(a): la superficie è molto simile, ma più regolare, rispetto a quella presente in Figura 4.5, rendendo quindi valide le stesse considerazioni fatte sull'altra.



(a) Superficie di volatilità locale generata dal modello A-H-K, al variare di T e di K/S_0.

(b) prezzi di mercato (pallini rossi) e prezzi ottenuti dalle modifiche di A-H-K, al variare di T e di K/S_0.

Figura 4.9: Risultati ottenuti utilizzando le modifiche proposte.

modello	Root Mean Squared Error	Leave-one-out	tempo
A-H-K	0.0292	0.0604	30.777 s

Tabella 4.2: Errore e tempo di esecuzione del modello Abasto-Hientzsch-Kust.

In Tabella 4.2 sono presenti l'errore medio e l'errore puntuale calcolato con il metodo Leave-one-out riferiti ai prezzi calcolati con Differenze Finite e con 10 sotto intervalli. Si è notato che il numero N_t nella formula (4.22) non influisce molto sull'esito qualora si utilizzino le Differenze Finite, poichè già nella calibrazione l'algoritmo tiene conto dei possibili errori dovuti alla presenza di numerosi sotto intervalli, compensandoli per trovare l'ottimo. In riferimento quindi ai dieci sotto intervalli si vede in Tabella 4.2 che gli errori sono maggiori rispetto a quelli ottenuti con il modello A-H, ma comunque inferiori al *bid-ask spread*, rendendo così anche questo un modello affidabile.

Per quanto riguarda il tempo di esecuzione in Tabella 4.2 invece, si ha un rallentamento abbastanza significativo rispetto al modello senza modifiche: andando ad analizzare più in dettaglio si scopre che è per la maggior

parte spiegato dall'approssimazione più fitta della derivata temporale, mentre l'ottimizzazione su $K\vartheta(K)$ non sembra influenzare la velocità del codice.

La questione cambia qualora si utilizzi il metodo Monte Carlo per ottenere i prezzi, come suggerito dagli autori. In Tabella 4.3 vengono confrontati il metodo originale e quello con le modifiche, utilizzando sempre Monte Carlo per il calcolo delle superficie dei prezzi. Sono stati usati i sotto intervalli nella calibrazione anche nel metodo A-H così da avere separati l'effetto dovuto all'ottimizzazione su ϑ o $K\vartheta(K)$ e quello dovuto alla griglia nel tempo più fitta.

	A-H		A-H-K	
sotto-interv.	RMSE	Leave-one-out	RMSE	Leave-one-out
0	0.3920	1.1120	0.1863	0.4867
10	0.2775	1.0226	0.0667	0.1469
50	0.3826	0.9927	0.0560	0.1354
100	0.3459	0.8783	0.0551	0.0941
500	0.2574	0.8372	0.0537	0.0922
1000	0.3497	0.8359	0.0478	0.0868

Tabella 4.3: Confronto tra i due metodi al variare del numero di sotto intervalli temporali.

Dal confronto si evince che, a differenza delle Differenze Finite, utilizzando il metodo Monte Carlo entrambe le modifiche proposte dagli autori hanno l'effetto di diminuire l'errore; si raggiunge quindi la conclusione che, a seconda del prodotto da prezzare e quindi dalla tecnica impiegata per ottenere il prezzo, conviene utilizzare o il caso base A-H oppure quello modificato A-H-K. Poichè tra gli obbiettivi di questa tesi vi è il prezzare al meglio le opzioni esotiche e con barriera, quindi necessariamente con le simulazioni Monte Carlo, il modello A-H-K è risultato il più adeguato a tale fine; per la comparazione con il modello parametrico di Gatheral e Jacquier perciò solo quest'ultimo verrà considerato.

In Figura 4.9(b) è possibile osservare i prezzi generati applicando tutte le modifiche suggerite dagli autori. Anche nel caso in esame si può notare che quasi tutti i pallini rossi vengono raggiunti dalla superficie; i prezzi generati sono crescenti sia nella variabile tempo che nella variabile strike, garantendo l'assenza di arbitraggio.

Capitolo 5

Il modello Gatheral-Jacquier (SSVI)

Tra i modelli parametrici per lo studio della volatilità implicita, lo *stochastic volatility inspired* o *SVI* sviluppato da Jim Gatheral e Antoine Jacquier in Merrill Lynch e successivamente presentato in [20] è sicuramente il più famoso ed utilizzato (indicato in questa tesi anche come G-J). I parametri del modello si riferiscono all'equazione che si ipotizza descriva la superficie di volatilità implicita; la caratteristica principale che lo distingue dagli altri modelli parametrici, per lo più governati da equazioni quadratiche o cubiche, è che presenta di un andamento asintoti- camente lineare e curvo nella parte centrale, ed è relativamente facile *fittare* i prezzi delle opzioni senza avere il Calendar Spread Arbitrage (guardare la sottosezione 5.2 per la definizione). Una volta stimati i parametri necessari per ottenere il fit, con poche modifiche e cambi di variabile è possibile ottenere la superficie di volatilità locale cercata, da utilizzare poi nella formula di Dupire (2.2).

In questo capitolo verrà esposto per sommi capi il modello SVI e poi più in dettaglio la sua modifica più importante, il **Surface Stochastic Volatility Inspired SSVI**; si metteranno in luce le problematiche dell'assenza di arbitraggio e infine le spiegazioni su come è stato implementato il SSVI in MATLAB.

Notazioni. Per ogni $K \in \mathbb{R}$ e $t > 0$, chiamiamo la volatilità implicita di Black&Scholes quella ottenuta invertendo i prezzi di mercato con $\sigma_{\text{BS}}(K,t)$ e definiamo la varianza implicita totale con

$$w(k,t) = \sigma^2_{\text{BS}}(K,t) \cdot t \tag{5.1}$$

dove si utilizza la trasformazione $K = F_t e^k$ (forward moneyness). La varianza implicita v può essere equivalentemente definita come $v(k,t) = \sigma^2_{\text{BS}}(K,t) =$

$w(k,t)/t$. Nel caso si fissi una *slice* temporale di scadenze, si utilizzerà la notazione $w(k,\chi)$ dove χ rappresenta l'insieme dei parametri che dipendono dalla parametrizzazione utilizzata e viene omessa la dipendenza dal tempo.

5.1 Spiegazione del modello

Esistono diverse versioni equivalenti della parametrizzazione SVI per descrivere la superficie implicita, ma le due più famose sono la raw e la natural parameterization.

La Raw SVI:

$$w(k,\chi_{\mathrm{R}}) = a + b\left\{\rho(k-m) + \sqrt{(k-m)^2+\sigma^2}\right\} \tag{5.2}$$

dove $a \in \mathbb{R}$, $b \geq 0$, $|\rho| < 1$, $m \in \mathbb{R}$, $\sigma > 0$ e la ovvia condizione $a + b\sigma\sqrt{1-\rho^2} \geq 0$ che assicura che $w(k,\chi_{\mathrm{R}}) \geq 0$ per ogni $k \in \mathbb{R}$. Il set di parametri è quindi $\chi_{\mathrm{R}} = \{a,b,\rho,m,\sigma\}$.

La Natural SVI:

$$w(k,\chi_{\mathrm{N}}) = \Delta + \frac{\omega}{2}\left\{1 + \zeta\rho(k-\mu) + \sqrt{\big(\zeta(k-\mu)+\rho\big)^2 + (1-\rho^2)}\right\} \tag{5.3}$$

dove $\Delta \in \mathbb{R}$, $\omega \geq 0$, $|\rho| < 1$, $\mu \in \mathbb{R}$ e $\zeta > 0$. Il set di parametri è quindi $\chi_{\mathrm{N}} = \{\Delta,\mu,\rho,\omega,\zeta\}$.

Proposizione 5. *[21] E' possibile passare in modo univoco tra una rappresentazione e l'altra attraverso le seguenti uguaglianze tra parametri. Dalla raw alla natural SVI:*

$$(a,b,\rho,m,\sigma) = \left(\Delta + \frac{\omega}{2}(1-\rho^2), \frac{\omega\zeta}{2}, \rho, \mu - \frac{\rho}{\zeta}, \frac{\sqrt{1-\rho^2}}{\zeta}\right), \tag{5.4}$$

e viceversa dalla natural alla raw SVI:

$$(\Delta,\mu,\rho,\omega,\zeta) = \left(a - \frac{\omega}{2}(1-\rho^2), m + \frac{\rho\sigma}{\sqrt{1-\rho^2}}, \rho, \frac{2b\sigma}{\sqrt{1-\rho^2}}, \frac{\sqrt{1-\rho^2}}{\sigma}\right). \tag{5.5}$$

Questi modelli hanno diverse proprietà interessanti. Prima di tutto dipendono solo da **5 parametri**, quindi possono essere facilmente e velocemente calibrati coi dati di mercato per ogni scadenza. Inoltre sono capaci di riprodurre correttamente il tipico Smile e, in accordo con quanto dimostrato dalla

formula di Roger Lee [31], mostrano un comportamento asintotico corretto, cioè lineare: $\sigma(k,T) \to \sqrt{k}$ per $k \to \pm\infty$. Infatti in riferimento alla parametrizzazione raw (5.2), l'asintoto destro in accordo con [20] è

$$w(k,\chi_{\mathrm{R}})_{\mathrm{r}} = a + b(1-\rho)(k-m), \tag{5.6}$$

mentre l'asintoto sinistro è

$$w(k,\chi_{\mathrm{R}})_{\mathrm{l}} = a - b(1+\rho)(k-m). \tag{5.7}$$

entrambi sono lineari nella moneyness quindi soddisfano la formula di Lee.

Si può inoltre dimostrare [22] che il modello SVI può essere visto come il limite dello Smile di volatilità del modello stocastico di Heston per $T \to \infty$.

Sempre in riferimento alla parametrizzazione raw (5.2) (ma le stesse considerazioni si possono fare anche sulla natural (5.3) grazie alla Proposizione 5) è possibile esplicitare i seguenti effetti dovuti a variazioni nei cinque parameri in ingresso:

- a: questo parametro è la costante della volatilità. Incrementando a si aumenta il livello generale della varianza; una traslazione verticale dello Smile.
- $b > 0$: questo parametro influenza la pendenza di entrambe le ali della volatilità. Aumentandolo quindi si restringe lo Smile.
- $\rho \in (-1,1)$: la correlazione ha come effetto una rotazione in senso anti-orario dello Smile; maggiore è ρ e maggiore (minore) è la pendenza dell'ala destra (sinistra).
- m: aumentando questo parametro lo Smile si sposta verso destra; una traslazione orizzontale dello Smile.
- $\sigma > 0$: l'ultimo parametro influisce negativamente sulla curvatura della parte centrale (at-the-money) dello Smile, corrispondente nella moneyness a $k = 0$. Incrementando σ si riduce la curvatura.

I modelli SVI replicano molto bene la superficie implicita per ogni maturity T fissata. Tuttavia mostrano il lato negativo durante la calibrazione: sono affetti dalla presenza di numerosi minimi locali, quindi diventano estremamente sensibili alla scelta dei dati iniziali [30]. Per risolvere il problema sono state introdotte diverse tecniche tra cui una riduzione dei parametri (da 5 a 2) [42] oppure la ricerca di numeri che diano risultati accettabili con alcune modifiche successive [45], ma in entrambi i casi non è stato comunque risolto il problema della possibile presenza di arbitraggio.

5.1.1 Il modello SSVI

Introduciamo ora la classe di modelli SVI chiamata SSVI (da 'Surface SVI') [21], che si presenta come un'estensione della natural parameterization (5.3).

Per ogni maturity $t \geq 0$, si definisce la varianza implicita totale at-the-money $\vartheta_t := \sigma^2_{BS}(0,t)t$. Dobbiamo però assumere che la funzione ϑ sia almeno di classe C^1 in $\mathbb{R}_+$. Un opzione ATM con maturity zero non ha valore quindi $\vartheta_0 = lim_{t\to 0}\vartheta_t = 0$.

Definizione 2 (SSVI)**.** *Sia φ una funzione da $\mathbb{R}_+$ a $\mathbb{R}_+$ tale che il limite $lim_{t\to 0}\vartheta_t\varphi(\vartheta_t)$ esista in $\mathbb{R}$. Il modello SSVI è caratterizzato dalla superficie definita da*

$$w(k,\vartheta_t) = \frac{\vartheta_t}{2}\left\{1 + \rho\varphi(\vartheta_t)k + \sqrt{\left(\varphi(\vartheta_t)k + \rho\right)^2 + (1-\rho^2)}\right\}. \tag{5.8}$$

Dal confronto con le altre forme della superficie di volatilità si nota che la SSVI non è nient'altro che la parametrizzazione natural SVI (5.3) con quintupla di parametri $\chi_N = \{0, 0, \rho, \vartheta_t, \varphi(\vartheta_t)\}$.

Va sottolineato il fatto che in questo modello si considera la superficie di volatilità in termini di ϑ_t (ATM variance time) e non di t (calendar time); questa modifica ha lo stesso effetto ed è stata pensata per l'utilizzo con lo stesso spirito con cui vengono utilizzati i **subordinatori**[1] nei processi stocastici [11].

Osservazione 2. *Nella parametrizzazione* (5.8)*, la curva della varianza ATM ϑ_t può essere vista come un (vettore) parametro della superficie di volatilità. Inoltre, questo parametro è direttamente osservabile dati i prezzi del mercato per un insieme finito di scadenze, e può essere considerato "well known" dai trader anche per scadenze che non sono esplicitamente quotate. Questo fa sì che, seppur considerato un parametro in quanto varia per ogni matrice di dati considerata, una volta che viene fissata si hanno tutti i valori per ogni t quindi ϑ_t non è più un parametro da stimare.*

[1] Molto brevemente i subordinatori sono particolari processi di Lèvy non decrescenti; servono principalmente per modellizzare il *business time*, cioè il fatto che nei mercati sono presenti sia momenti di frenesia e agitazione, più veloci, sia momenti più tranquilli e lenti. Per fare ciò viene sostituita la dipendenza dal tempo (lineare per definizione) nel modello matematico utilizzato, con un processo stocastico avente proprietà specifiche, così da riprodurre meglio l'imprevedibilità dell'alternanza *frenesia-calma* rispetto all'utilizzo del tempo classico lineare. Possono essere utilizzati sia nei moti browniani geometrici, sia nei processi con salto più complicati e sia in modelli per la volatilità stocastica (con o senza salti).

Per quanto riguarda la forma della funzione φ, sono possibili diverse parametrizzazioni, ognuna con diverse caratteristiche:

Heston-like parameterization

$$\varphi(\vartheta) \equiv \frac{1}{\lambda\vartheta}\Big\{1 - \frac{1 - e^{-\lambda\vartheta}}{\lambda\vartheta}\Big\} \quad con\ \lambda > 0. \tag{5.9}$$

Power-law parameterization

$$\varphi(\vartheta) \equiv \eta\vartheta^{-\gamma} \quad con\ \eta > 0\ \wedge\ 0 < \gamma < 1. \tag{5.10}$$

Generalized parameterization generalizzazione del SVI sviluppata in [24], nella quale oltre ad utilizzare uno dei due casi precedenti per la φ, la superficie è definita da

$$w(k,t) = \vartheta_{\mathrm{t}}\Psi(k\varphi(\vartheta_{\mathrm{t}})) \tag{5.11}$$

dove Ψ rappresenta la generica forma che può assumere l'equazione che descrive la superficie.

5.2 Assenza di arbitraggio

In questa sezione viene data una definizione modello-indipendente dell'arbitraggio (statico), per poi dare delle condizioni più restrittive a seconda del modello utilizzato (gli SVI o il SSVI). Di solito le definizioni di arbitraggio statico sono fatte in relazione al prezzo delle opzioni vanilla, in questo caso invece viene espresso direttamente in termini di volatilità.

Definizione 3. *Una superficie di volatilità è priva di arbitraggio statico se e solo se le seguenti condizioni sono soddisfatte:*

1. *è priva di Calendar Spread Arbitrage;*
2. *ad ogni slice temporale è priva di Butterfly Arbitrage.*

Il Calendar Spread Arbitrage è principalmente espresso come la *non* monotonicità dei prezzi di un'opzione Call europea rispetto alla maturity: Matematicamente, la superficie w non presenta calendar spread arbitrage se e solo se

$$\partial_{\mathrm{t}} w(k,t) \geq 0, \quad \forall k \in \mathbb{R}\ \wedge\ t > 0.$$

L'assenza del Butterfly Arbitrage invece corrisponde all'esistenza di una misura di martingala priva di rischio e alla classica definizione di assenza

arbitraggio statico. Qui viene considerato solo un istante temporale della superficie implicita, cioè la mappa $k \mapsto w(k,t)$ per una maturity fissata $t > 0$, e assumiamo che essa sia almeno di classe $C^2(\mathbb{R})$ per ogni $t \geq 0$.

Si definisce la funzione $g\colon \mathbb{R} \mapsto \mathbb{R}$ come

$$g(k) := \left(1 - \frac{kw'(k)}{2w(k)}\right)^2 - \frac{w'(k)^2}{4}\left(\frac{1}{w(k)} + \frac{1}{4}\right) + \frac{w''(k)}{2}. \tag{5.12}$$

Una slice è detta priva di butterfly arbitrage se la corrispondente densità è non-negativa, quindi se e solo se $g(k) \geq 0$ per ogni $k \in \mathbb{R}$ e $\lim_{k\to+\infty} d_+(k) = -\infty$ (dove $d_+(k) = -k/\sqrt{w(k)} + \sqrt{w(k)}/2$) [21].

Nel caso dei modelli di superficie di volatilità SSVI (5.8), questi due possibili arbitraggi vengono evitati grazie a due teoremi che forniscono due condizioni necessarie e sufficienti per garantirne l'assenza.

Teorema 1. *[21] La superficie SSVI* (5.8) *è priva del* ***Calendar Spread Arbitrage*** *se e solo se*

1. $\partial_{\mathrm{t}}\vartheta_{\mathrm{t}} \geq 0, \quad \forall t \geq 0$;
2. $0 \leq \partial_\theta(\vartheta\varphi(\vartheta)) \leq \frac{1}{\rho^2}\left(1 + \sqrt{1-\rho^2}\right)\varphi(\vartheta), \quad \forall\theta > 0$

dove il bordo superiore è infinito quando $\rho = 0$.

La condizione 1. del teorema è quella alla quale si è fatto riferimento nell'Appendice A e che ha influenzato la scelta del sottostante S&P500, infatti si è dovuto scegliere un indice nel quale i valori ATM della volatilità implicita fossero non decrescenti, o poco decrescenti in modo tale che una volta moltiplicati per il tempo t risultassero crescenti.

Teorema 2. *[21] La superficie SSVI* (5.8) *è priva del* ***Butterfly Arbitrage*** *se le seguenti condizioni sono soddisfatte per ogni* $\vartheta > 0$*:*

1. $\vartheta\varphi(\vartheta)(1 + |\rho|) < 4$;
2. $\vartheta\varphi(\vartheta)^2(1 + |\rho|) \leq 4$;

Per lo svolgimento della tesi si è scelto di utilizzare la Power-law parameterization, andando a influenzare le condizioni che dipendono dalla φ come la condizione 2 del Teorema 1 e quelle del Teorema 2: per soddisfarle deve valere $\gamma \in [0,1]$; come spiegato però in [21] si possono ottenere gli stessi risultati anche con le altre parametrizzazioni.

Considerando la formula (5.10) si può far vedere che per $\eta > 0$ e $0 < \gamma < 1$

$$\frac{\partial_\vartheta(\vartheta\varphi(\vartheta))}{\varphi(\vartheta)} = 1 - \gamma \in (0, 1)$$

vale per ogni $\vartheta > 0$, entrambe le condizioni del primo teorema perciò sono soddisfatte.

Sempre in riferimento alla Power-law parameterization (5.10), nell'articolo [21] viene mostrato come la superficie può essere affetta da arbitraggio solo per scadenze molto lunghe, cioè nel caso $\gamma = 1/2$ le condizioni valgono solo fino a un certo istante di tempo t^*. Tuttavia il comportamento per grandi ϑ può essere facilmente modificato così da garantire sempre l'assenza di arbitraggio statico. Se si considera infatti l'alternativa

$$\varphi(\vartheta) = \frac{\eta}{\vartheta^\gamma(1+\vartheta)^{1-\gamma}} \tag{5.13}$$

la superficie è priva di butterfly arbitrage se è soddisfatta la condizione $\eta(1+|\rho|) \leq 2$.

Teorema 3. *[21] Sia $(k,t) \mapsto w(k,t)$ una superficie di volatilità SSVI (5.8) che soddisfi tutte e quattro le condizioni dei due precedenti teoremi, e sia $a\colon \mathbb{R}_+ \to \mathbb{R}_+$ una funzione nel tempo non negativa e crescente. Allora la superficie di volatilità $(k,t) \mapsto w_{\mathrm{a}}(k,\theta_{\mathrm{t}}) := w(k,\theta_{\mathrm{t}}) + a_{\mathrm{t}}$ è anch'essa priva di arbitraggio statico.*

Il teorema appena esposto ci tornerà utile nell'implementazione quando bisognerà aggiungere una sorta di "penalità" nel caso in cui due slice adiacenti si intersechino.

5.3 Dalla SSVI alla Volatilità Locale

Una caratteristica importante dei modelli SVI che va messa sotto i riflettori per poter capire le differenze con gli altri modelli è la seguente: oltre al fatto che viene assunta una forma parametrica a priori per la superficie, i parametri vengono calibrati in modo tale da ottenere una superficie di volatilità implicita, non locale come in Andreasen-Huge [4]. Avendo già esposto nel Capitolo 2 i motivi e le situazioni nelle quali è meglio utilizzare la superficie locale rispetto a quella implicita, il passaggio successivo una volta calibrata tutta la superficie implicita è quello di trovare una formula per passare da essa alla volatilità locale.

Prima di spiegare questa formula bisogna fare alcune considerazioni. Partendo dalla ormai nota equazione di Dupire (2.2) è possibile ottenere la volatilità locale invertendo la formula, come mostrato nell'equazione (2.8); questa però ovviamente presuppone che i prezzi C siano presenti per ogni valore di (K, T) così da poterne calcolare tutte le derivate necessarie. Non è possibile quindi usare direttamente la (2.8) ma bisogna interpolare ed estrapolare i prezzi dal mercato e poi approssimare numericamente le derivate. Inoltre tale procedura è piuttosto sensibile alle instabilità numeriche e ad errori: particolarmente critica è la derivata seconda $\frac{\partial^2 C}{\partial K^2}$ al denominatore moltiplicata per K^2, che ne amplifica l'errore, dovuto al fatto che questa derivata può essere molto piccola per opzioni molto in-the-money o out-the-money, producendo addirittura varianze negative.

Avendo a disposizione dal mercato le volatilità implicite σ_{BS} e non i prezzi, un passaggio abbastanza naturale è quello di riformulare l'equazione inversa di Dupire (2.8) in termini di questi valori impliciti; in [18] [12] l'idea è quella di inserirvi la formula di Black&Scholes e le sue derivate assumendo che

$$C = C_{\text{BS}}(S_0, K, T, \sigma_{\text{BS}}(K, T))$$

così che applicando la trasformazione *moneyness* ed esprimendo l'equazione di Dupire (2.2) in termini della varianza implicita totale w definita in (5.1) è possibile [19] ottenere la formula che dalla volatilità implicita restituisce la volatilità locale:

$$\sigma^2_{\text{LV}} = \frac{\frac{\partial w}{\partial T}}{1 - \frac{K}{w}\frac{\partial w}{\partial K} + \frac{1}{4}\big(-\frac{1}{4} - \frac{1}{w} + \frac{K^2}{w^2}\big)\big(\frac{\partial w}{\partial K}\big)^2 + \frac{1}{2}\frac{\partial^2 w}{\partial K^2}}. \tag{5.14}$$

Si rimanda all'appendice D per la spiegazione su come è stata ottenuta la formula (5.14).

Sebbene questa formula sia più complessa rispetto alla formula (2.8), dal punto di vista pratico è certamente più utilizzabile: la derivata seconda rispetto a K al denominatore è ancora presente, ma ora è solo uno dei termini di una somma; errori piccoli nell'approssimazione della derivata seconda quindi non si traducono necessariamente in errori grandi nella valutazione della superficie di volatilità locale.

A questo punto entrano in gioco i modelli SVI. Come si può vedere anche la formula (5.14) richiede una superficie implicita senza buchi e continua

che sul mercato non è possibile osservare[2]. Verrà spiegato nell'implementazione come è possibile ottenere questa superficie implicita continua, per ora la consideriamo già ottenuta per poter andare avanti nel ragionamento e la indichiamo con σ_{SVI}, e la utilizziamo per poter ricavare la superficie di volatilità locale.

Attenendosi a quanto esposto in [19], la formulazione più generale possibile per il metodo Stochastic Volatility Inspired è

$$w_{\text{SVI}}(k,t) = \alpha_1 + \alpha_2\Big[\alpha_3(k-\alpha_4) + \sqrt{(k-\alpha_4)^2+\alpha_5}\Big] \tag{5.15}$$

dove non c'è nessuna restrizione sulla parametrizzazione $A = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5\}$ e supponiamo α_{i} già calibrati con i dati di mercato.

Per poter ricostruire la superficie di volatilità locale a questo punto si può notare che l'equazione (5.14) richiede i valori di w, $\frac{\partial w}{\partial K}$,$\frac{\partial^2 w}{\partial K^2}$ e $\frac{\partial w}{\partial T}$. Considerando l'equazione (5.15), per ogni $k \in \mathbb{R}$ e per $t \in [T_{\text{n}}, T_{\text{n+1}}]$ abbiamo[3]:

$$\begin{aligned} w = w(k,t) &= \tau_{\text{n}} w(k,T_{\text{n}}) + \tau_{\text{n+1}} w(k,T_{\text{n+1}}),\\ \frac{\partial w}{\partial t} &= \frac{1}{T_{\text{n+1}} - T_{\text{n}}}\Big[w(k,T_{\text{n+1}}) - w(k,T_{\text{n}})\Big],\\ \frac{\partial w}{\partial k} &= \tau_{\text{n}} f_{\text{n}}(k) + \tau_{\text{n+1}} f_{\text{n+1}}(k),\\ \frac{\partial^2 w}{\partial k^2} &= \tau_{\text{n}} g_{\text{n}}(k) + \tau_{\text{n+1}} g_{\text{n+1}}(k). \end{aligned} \tag{5.16}$$

dove

$$f_{\text{n}}(k) = \alpha_2^n \alpha_3^n + \frac{\alpha_2^n\big(k-\alpha_4^n\big)}{\sqrt{(k-\alpha_4^n)^2+\alpha_5^n}},$$

[2]In questo caso si richiede che la superficie che debba essere almeno C^1 sia quella delle volatilità implicite anziché quella dei prezzi, quindi a prima vista sembra che il problema sia stato semplicemente spostato e non risolto. In realtà ragionando in termini di superficie implicita è possibile utilizzare i modelli SSVI per ottenere tale superficie *piena*, per poi passare attraverso una formula alla superficie locale, e quindi da essa alla superficie dei prezzi, che risulterà $\in C^1$.

[3]Indichiamo con t una maturity generica, compresa tra un istante temporale generico T_{n} e il suo successivo presenti inizialmente nella griglia della volatilità implicita. Questa formula ci garantisce quindi, iterando su tutta la superficie, l'estrapolazione di tutti i valori possibili nella variabile tempo; riuscendo a colmare i vuoti inizialmente presenti.

$$g_{\mathrm{n}}(k) = \frac{\alpha_2^n}{\sqrt{(k-\alpha_4^n)^2+\alpha_5^n}} - \frac{\alpha_2^n(k-\alpha_4^n)^2}{[(K-\alpha_4^n)^2+\alpha_5^n]^{3/2}},$$
$$\tau_{\mathrm{n}} = \frac{T_{\mathrm{n+1}}-t}{T_{\mathrm{n+1}}-T_{\mathrm{n}}},$$
$$\tau_{\mathrm{n+1}} = \frac{t-T_{\mathrm{n}}}{T_{\mathrm{n+1}}-T_{\mathrm{n}}}.$$

Sebbene la formula (5.14) sia meno sensibile agli errori di approssimazione rispetto alla (2.8), può succedere per costruzione che il numeratore, o più facilmente il denominatore, diventino negativi. Questo può succedere a causa dell'instabilità numerica comunque non eliminabile, dovendo così ricorrere ad un espediente [12]: durante l'implementazione può essere utile aggiungere una piccola costante positiva ϵ sia al numeratore sia al denominatore, evitando così il problema di ottenere varianza negativa e quindi valori complessi della volatilità locale.

Definendo il numeratore e il denominatore rispettivamente $N(K,T)$ e $D(K,T)$ si può considerare l'equazione

$$\sigma_{\mathrm{LV}}(K,T) = \sqrt{\frac{N(K,T)+\epsilon_1}{D(K,T)+\epsilon_2}}\ . \tag{5.17}$$

5.4 Implementazione

In questa sezione vengono esposti gli algoritmi utilizzati per implementare il modello G-J e i relativi commenti ad essi. Prima di ogni nuovo algoritmo è presente una breve descrizione, così da poter essere consultata velocemente durante la lettura della parte riguardante la funzione MAIN.

Algoritmo 9: GJ - Main Function

```
/* - parte 1 -                                                    */
Inizzializza le variabili e la memoria necessaria;
Dati: vettore K degli strike
Dati: vettore T degli istanti temporali (le diverse maturity dei
      prodotti)
foreach Istante temporale→ i do
    if si trova nel primo istante then
        non considera l'incremento;
    else
        Lunghezza del passo(i) ← T(i) − T(i − 1);
    end;
end;
Dati: vettore Fwd dei prezzi forward del sottostante
scritti come percentuale del valore del sottostante a t = 0;
foreach Istante temporale → i do
    k(i) ← log{K/Fwd(i)} ;
    modifica gli strike per la Moneyness;
end;
Dati: Matrice ImpVol formata dalle volatilità implicite prese dal
      mercato
foreach Istante temporale → i do
    foreach Strike → k do
        Prezza un opzione Call ←
        B&S(T = i, strike = k, volat = ImpVol(i, k));
        Riempie la matrice R (chiamata prezzi di mercato);
    end;
end;
/* - parte 2 -                                                    */
Theta ← impVol(100%, :)^2 · T;
considera solo la riga ATM della matrice e la modifica per poter essere
usata in SSVI;
```

foreach $\eta \in [0, 2]$ **do**
  foreach $\rho \in [-1, 1]$ **do**
    calcola tutti i parametri della superficie in una volta sola usando (η, ρ) *come guess iniziale e ne considera l'errore*;
    [$\sim$,fval] $\leftarrow$ `runnestedTotal(`*T,Theta, k, impVol, param*$\leftarrow(\eta, \rho)$`)`;
    min_rho $\leftarrow \rho$ corrispondente al punto con errore minimo;
    min_eta $\leftarrow \eta$ corrispondente al punto con errore minimo;
  end;
end;

`/* - parte 3 -` `*/`
paramSSVI $\leftarrow$ matrice 3×26 contenente i parametri ottenuti da (min_eta,min_rho) come guess iniziale;
calcola tutta la superficie W utilizzando i parametri appena calcolati;
[w,phi] $\leftarrow$ `SSVI(`*k, Theta, paramSSVI*`)`;
phi $\leftarrow$ scalare che segue da Power-law parameterization;
alpha $\leftarrow$ vettore 1×5 contenente i 5 coefficienti della SVI;

`/* - parte 4 -` `*/`
for *i = seconda* **to** *ultimascadenza* **do**
  Inizializza i parametri e gli alpha con quelli all'istante precedente (con suffisso -Ex) ;
  `/* - parte 4.1 -` `*/`
  x $\leftarrow$ `Runnested(`*Theta(i), k(:,i), impVol(:,i), paramSSVI, paramEx*`)`;
  calcola i η *e* ρ *solo per l'istante i*;
  [w,phi] $\leftarrow$ `SSVI(`*k(:,i), Theta(i), x*`)`;
  calcola la funzione di volatilità (total implied variance) con i parametri appena stimati;
  paramSSVI(:,i) $\leftarrow$ x;
  foreach *elemento di alpha* **do**
    Ricalcola basandosi sui nuovi parametri $paramSSVI(:, i)$, *phi* e $Theta(i)$;
  end;
  `/* - parte 4.2 -` `*/`
  Inizializza la matrice $surfaceLocal$ 10 volte più fitta in T;

... continua dal ciclo for precedente;
    foreach *istante*→ *j tra i e i* + 1 **do**
        calcola $w(k, j)$ interpolando con pesi τ_{n} e $\tau_{\mathrm{n+1}}$;
        Calcola la derivata di w in T;
        Calcola la derivata di w in X;
        Calcola la derivata seconda di w in X;
        N ← numeratore della formula (5.14);
        D ← denominatore della formula (5.14);
        if $\min(N) < 0$ **then**
            N ←N + ϵ_1;
        end;
        if $\min(D) < 0$ **then**
            D ← D + ϵ_2;
        end;
        $surfaceLocal(i, j) = \sqrt{\frac{N}{D}}$;
    end;
end;

`/* - parte 5 - */`

suface_l ← estrae dalla matrice $surfaceLocal$ le colonne associate alla griglia di partenza;

... viene calcolata tutta la superficie dei prezzi a partire dalla volatilità locale surface_l o tramite differenze finite o tramite Monte Carlo e viene salvata in prezziTOT;

`/* - parte 6 - */`

for *i* = *prima* **to** *ultimascadenza* **do**
    for *j* = *primo* **to** *ultimoStrike* **do**
        if *l'elemento della matrice ImpVol corrispondente a scadenza i e strike j è* $\neq 0$ **then**
            err ← somma errori $(R(j, i) - prezziTOT(j, i))$ al quadrato;
        end
    end
end;
err ← calcola l'errore quadratico medio;

Oltre alle volatilità, in questo modello vengono presi in input i dati forward del prezzo del sottostante alle date di scadenza in linea con gli altri dati scaricati dal provider. Questi prezzi non sono nient'altro che la stima, la previsione del valore ATM che assumerà il titolo nelle date future, e vengono utilizzati nella trasformazione *Moneyness* dove i nuovi strike vengono

espressi come $k_{\mathrm{i,j}} = \log\left(Fwd_{\mathrm{i}}/K_{\mathrm{j}}\right)$ dove il logaritmo serve per linearizzarli e centrarli in 0.

Anche in questo caso la matrice delle volatilità scaricata è stata successivamente modificata per rispondere alle esigenze di implementazione e sono stati creati i prezzi di mercato.

Il vettore Theta rappresenta la varianza implicita totale ATM $\vartheta_{\mathrm{t}} := \sigma^2_{\mathrm{BS}}(0,t)\cdot t$, cioè il quadrato della volatilità corrispondente allo strike 100% moltiplicata per la scadenza[4].

Nella PARTE 2 vengono anche scelti i valori iniziali dell'ottimizzazione, sui quali bisogna fare una piccola parentesi. I modelli parametrici e soprattutto l'SVI proposto da Gatheral-Jacquier sono molto instabili e grazie alla presenza di numerosi minimi locali si possono ottenere risultati molto discordanti partendo da valori relativamente vicini, o ottenere addirittura risultati errati come volatilità negative. A differenza dei modelli precedenti, dove si erano utilizzate come guess iniziali le volatilità di mercato, qua non è possibile utilizzare lo stesso meccanismo perché vengono ottimizzati i parametri della formula (5.8) e non la volatilità che se ne ottiene, quindi serve una guess iniziale per i valori dei parametri, non estraibile a priori dai dati di mercato.

Per poter scegliere i due valori iniziali per η e ρ si è quindi deciso di creare una griglia equispaziata di 30 intervalli, dove ρ varia in $[-1,1]$ e η in $[0,2]$ (valori dovuti alla condizione per l'assenza di arbitraggio (5.13)), e di passare alla funzione RUNNESTEDTOTAL i parametri iniziali ogni volta differenti, così da far calcolare al codice tutti i parametri della superficie e vedere dove si ottiene il minimo scarto tra le volatilità implicite scaricate e quelle calcolate dal modello SSVI tramite la formula (5.8). In Figura 5.1 è rappresentato l'errore compiuto dall'ottimizzatore di MATLAB al variare dei parametri iniziali; si può osservare la presenza di numerosi picchi e aree di forte instabilità. Il minimo di questa superficie è 0.1411, al quale corrispondono i parametri $[\eta = 0.9655, \rho = -0.1724]$ che verranno quindi utilizzati come punto di partenza per l'ottimizzazione slice-by-slice.

Nella PARTE 3 vengono inizializati anche gli altri parametri utilizzati come guess iniziale. *paramSSVI* è una matrice di dimensione 3×26 contenente i parametri che andranno successivamente stimati step-by-step. La dimensione è data dal fatto che per ogni scadenza sono necessari due parametri[5] per poter stimare le curve w della formula (5.8), e precisamente η (che è all'interno della

[4] volendo essere più precisi bisognerebbe quindi chiamarla ϑ_{T} in quanto viene calcolata solo alle scadenze ma per utilizzare le stesse notazioni dell'articolo di riferimento indicheremo il tempo generico t.

[5] il terzo, γ, è sempre uguale a $1/2$ per le proprietà della parametrizzazione Power-law.

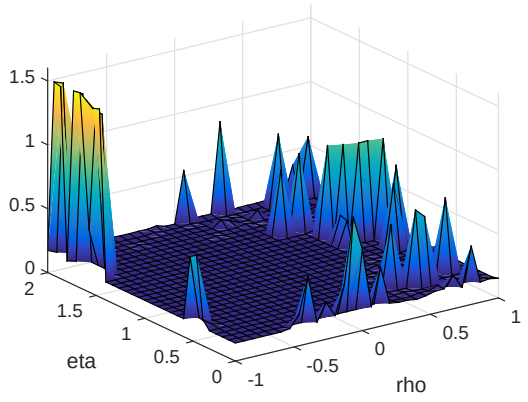


Figura 5.1: Errore dell'ottimizzazione globale al variare dei parametri iniziali.

funzione φ) e il ρ. Si ricorda che ϑ_t è preso dal mercato quindi non rientra nei parametri da stimare.

alpha è un vettore di dimensione 5 nel quale vengono salvati i valori dei parametri della formulazione più generale (5.15) necessari per passare dalla volatilità implicita SSVI alla volatiltià locale.

Una spiegazione più dettagliata va fatta circa la forma dei componenti del vettore *alpha*: quando è stato introdotto il modello SSVI della varianza implicita totale (5.8), si è aggiunto che poteva essere visto come la parametrizzazione *natural SVI* (5.3) con quintupla di parametri $\chi_N = \{0, 0, \rho, \vartheta_t, \varphi(\vartheta_t)\}$. Tuttavia la formula (5.15) è la generalizzazione della versione *raw* (5.2), si è dovuta quindi utilizzare l'equazione che collega una rappresentazione all'altra (5.14) per ottenere i parametri α_i espressi tramite i parametri stimati dall'ottimizzazione:

$$\begin{aligned} \chi_{\text{SSVI}} \longleftrightarrow \chi_N &= \{0, 0, \rho, \vartheta, \varphi(\vartheta)\} \\ (5.15) \sim \chi_{\text{raw}} &\{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5\} \end{aligned} \tag{5.18}$$

quindi utilizzando la (5.14)

$$\big(\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5\big) = \Big(\frac{\vartheta}{2}\big(1-\rho^2\big), \frac{\vartheta\varphi(\vartheta)}{2}, \rho, \frac{-\rho}{\varphi(\vartheta)}, \frac{\sqrt{1-\rho^2}}{\varphi(\vartheta)}\Big). \tag{5.19}$$

La funzione RUNNESTEDTOTAL ricerca i parametri che minimizzino la distanza tra le volatilità create dal modello e quelle presenti sul mercato[6]; si

[6]Si ricorda che grazie alla biunivocità tra volatilità implicite e prezzi generati con B&S, minimizzare la distanza tra le prime garantisce gli stessi risultati che minimizzando i secondi.

rimanda alla spiegazione della funzione RUNNESTED il contenuto ed i passaggi con i quali questa minimizzazione è stata eseguita; l'unica differenza tra le due funzioni sta nel fatto che nella -Total vengono stimati in una volta sola tutti i 2×26 parametri del modello, mentre nell'altra viene fatto slice-by-slice. Lo scopo di aver creato la funzione RUNNESTEDTOTAL deriva dal fatto che, come suggerito dagli stessi autori, si è fatta un'ottimizzazione su tutta la superficie, per poi utilizzare i valori trovati come guess iniziale per il modello vero e proprio, che lavora un istante di tempo alla volta. Una volta calcolati questi parametri, la funzione SSVI calcola i $w(k,t)$ e le φ, queste ultime necessarie per aggiornare i vettori *alpha*.

Nella PARTE 4 vengono:

- salvati i parametri stimati e il vettore *alpha* dell'istante precedente necessari per il calcolo delle derivate nella PARTE 4.2;
- Risolto tramite la funzione RUNNESTED il problema di ottimizzazione vincolata;
- Dato questo valore in input alla funzione SSVI, la quale restituisce il vettore dei valori della superficie implicita corrispondenti all'istante i-esimo;
- Calcolati i nuovi valori dei componenti del vettore *alpha* sulla base delle nuove stime di φ e ρ;
- Costruita la superficie di volatilità locale a partire da quella implicita SSVI seguendo la formula (5.14).

La funzione RUNNESTED restituisce il vettore contenente i parametri η, ρ e γ, partendo da quelli stimati da RUNNESTEDTOTAL e minimizzando la distanza tra le volatilità implicite del mercato e quelle generate con i parametri stimati più una certa penalizzazione. I valori appena stimati vengono a loro volta dati in input alla funzione SSVI che in questo punto del codice ha lo scopo di calcolare la varianza implicita totale secondo la formula (5.8) per tutti gli strike k ma per una sola slice, così da poter aggiungere una fetta alla superficie di volatilità implicita. Avendo a questo punto ottenuto tutti i parametri η,ρ e φ caratterizzanti la maturity i, è possibile aggiornare i cinque valori α per poter creare la rappresentazione SVI generale e passare così alla volatilità locale.

Nella PARTE 4.2 viene creata la superficie di volatilità locale partendo dalla implicita creata con SSVI. Questo passaggio è fondamentale per poter

utilizzare la superficie nell'equazione di Dupire (2.2) ed avvalorarsi di tutte le proprietà che la volatilità locale possiede già esposte nel Capitolo 2.

Va premesso il fatto che nella formula (5.14) per passare da una volatilità all'altra è già stata considerata la possibilità di calcolare i valori anche per t compresi tra due maturity inizialmente presenti: infatti osservando le formule che seguono la (5.14) si vede che si compie una interpolazione assegnando determinati pesi alla volatilità della scadenza precedente e a quella successiva del t considerato. In questo caso viene iterato il processo per 10 punti compresi tra t_{i} e $t_{\mathrm{i+1}}$ $\forall i$, in ognuno dei quali viene calcolata la volatilità locale tramite (5.14).

Per ognuno di questi sotto intervalli sono state considerate due slice temporali e calcolate quindi le derivate nel tempo e negli strike tipiche dell'equazione di Dupire (2.2).

Nella PARTE 5, avendo a questo punto a disposizione tutta la superficie di volatilità locale, si creano i prezzi associati a tali volatilità per confrontarli con i prezzi di mercato per capire quanto il modello Gatheral-Jacquier produca risultati accurati ed utilizzabili a livello pratico. In riferimento all'articolo dove è stato preso il modello di questo capitolo, i due autori non hanno suggerito alcun modo specifico col quale creare i prezzi[7] quindi sono lasciati abbastanza al lettore gli step successivi, a seconda della necessità. Dato che si è già analizzato nei capitoli passati come ottenere i prezzi dalla volatilità locale e dall'equazione di Dupire (2.2), la scelta più ovvia è stata quella di prezzare utilizzando sia il metodo delle differenze finite sia il metodo Monte Carlo, così da "riutilizzare" le stesse considerazioni fatte in precedenza ed avere un livello di confronto dei risultati maggiore in quanto prodotti con la stessa tecnica. Si rimanda quindi all'implementazione del capitolo precedente per la spiegazione di questi passaggi.

Nella parte conclusiva del codice viene calcolato l'errore quadratico medio del modello. Anche in questo modello questo calcolo viene eseguito solo sui prezzi di mercato, cioè quelli corrispondenti a volatilità non nulle.

[7] anche perché il modello spiegato si riferisce esclusivamente alla volatilità implicita.

FUNZIONE RUNNESTED

La funzione Runnested risolve il problema dell'ottimizzazione vincolata per la determinazione dei parametri η,γ e ρ in ogni istante temporale (o globalmente se si sta utilizzando runnestedTotal).

Algoritmo 10: GJ - Ottimizzazione Vincolata

Function [x,fval] = `Runnested` (Theta, forward strike → k, impVol, valore iniziale → x0, parametri del passo precedente→paramEx, vola del passo precedente→Wvecchio)

Viene definita la funzione obbiettivo;
[x,fval] = fmincon(@(x) Nestedfun(Theta,k,impVol,x,paramEx, Wvecchio), x0 ,@confun);

Function f = Nestedfun(x,Theta,k,impVol,paramEx, Wvecchio)

eta ← primo elemento di x;
rho ← secondo elemento di x;
gamma ← $\frac{1}{2}$;

W ← `SSVI(`*k(:,i), Theta(i), parametri←[eta, rho, gamma]*`)`;
calcola la funzione W per ogni strike k presente nella griglia;

Cross ← `Crossedness(`*W, Wvecchio, k, Theta(i), Theta(i-1), parametri, paramEx*`)`;
controlla la presenza di intersezioni tra la W appena calcolata e quella all'istante temporale precedente Wvecchio;

W ← $\sqrt{\frac{W}{T}}$;
dalla varianza implicita totale ritorno alla volatilità per poterla confrontare con impVol;

foreach *elemento di W* **do**
    $sum \leftarrow sommaerrori(W(i) - impVol(i))^2$;
end;
f ← $\sqrt{sum} + 100 * Cross$;
penalizzazione dell'incrocio di due W consecutive di un fattore 100;
% calcola la distanza tra le vola reali ImpVol e quelle W generate dal modello;
end;
end;

La funzione NESTEDFUN produce per ogni vettore di parametri x la total implied variance w generata tramite la funzione SSVI per un t fissato, la quale viene passata alla funzione CROSSEDNESS insieme ai parametri e alla w calcolati in $t-1$; si rimanda la spiegazione della funzione CROSSEDNESS più avanti.

Una volta ottenute le penalizzazioni e avendo invertito la formula (5.1) per riscrivere la curva w dimensionalmente come una volatilità, si calcola la distanza tra essa e quella di mercato $impVol$; restituendo questa distanza e minimizzandola si ottengono i parametri che, qualora utilizzati nella funzione (5.8), generino la curva migliore per approssimare la volatilità scaricata.

VINCOLI

La funzione Confun contiene i vincoli utilizzati nella funzione Runnested.

Questi impongono che ogni elemento da minimizzare (nel nostro caso gli eta e rho che servono per comporre la funzione $w(k,t)$) rispetti determinati vincoli: ρ dev'essere in modulo minore di 1, η deve assumere solo valori positivi, ed entrambi devono relazionarsi tramite la condizione di assenza di arbitraggio $\eta\big(1+|\rho|\big) \leq 2$ (condizione dell'assenza del *butterfly arbitrage* specifica per la Power-law parameterization).

La funzione CONFUNTOTAL esegue esattamente lo stesso compito ma in maniera vettoriale per la funzione RUNNESTEDTOTAL.

FUNZIONE CROSSEDNESS

Partendo dalla *guess* iniziale globale sviluppata in RunnestedTotal, il modello G-J ripercorre la superficie della varianza implicita totale calibrando di nuovo i parametri slice-by-slice, considerando durante la calibrazione una forte penalità nel caso si presenti il Calendar Spread Arbitrage (cioè si incrocino le linee nel *plot* della varianza totale); la funzione Crossedness calcola questa penalità.

Algoritmo 11: GJ - Crossedness

Function Cross = `Crossedness` (w, k, Theta(i), Theta(i-1), param, paramEx)

%Analizza due a due istanti di tempo della superficie di volatilità e controlla se si incrociano;
for $i = seconda$ **to** *ultimascadenza* **do**
  vettore $Now \leftarrow w(:, i)$;
  vettore $Old \leftarrow w(:, i-1)$;
  while *trovo* 4 *intersezioni* **do**
    Considero la curva che inizia più in basso tra Old e Now;
    Cerco il primo punto dove $(Old - Now)$ cambia segno;
    if *non trovo nessun punto* **then**
      Ferma la ricerca;
    else
      Salva lo Strike in questione;
      Cerca il successivo;
    end;
  end;
  ordina le intersezioni;
  if *trovata solo una* **then**
    $\widetilde{k_1} \leftarrow k - 1$;
    $\widetilde{k_2} \leftarrow k + 1$;
  else
    $\widetilde{k_1} \leftarrow k_1 - 1$;
    $\widetilde{k_{\mathrm{i}}} \leftarrow$ media tra $k_{\mathrm{i}-1}$ e k_{i} ;
    $\widetilde{k}_{\mathrm{n}+1} \leftarrow k_{\mathrm{n}} + 1$;
  end;
  foreach *intersezione Trovata* **do**
    $c_{\mathrm{i}} \leftarrow$
    $\max\left[0; \texttt{SSVI}(\widetilde{k_{\mathrm{i}}}, Theta(i), \chi_1) - \texttt{SSVI}(\widetilde{k_{\mathrm{i}}}, Theta(i-1), \chi_2)\right].$;
  end;
  Cross $\leftarrow \max(c_{\mathrm{i}})$;
  ritorna la massima differenza tra una curva e la successiva nei punti più lontani dalle intersezioni;
end;
end;

Per calcolare questa penalità per ogni istante di tempo, si considerino due slice SSVI con parametri differenti χ_1 e χ_2 con $t_2 > t_1$. Vengono calcolati

anzitutto i punti $k_{\mathrm{i}}(i = 1, \ldots, n)$ con $n \leq 4$ dove le linee si incrociano, ordinandoli poi in ordine crescente. E' stato dimostrato da Cardano e Ferrari [9] che posso esistere al massimo 4 intersezioni tra due curve che seguono la formula (5.8), infatti se si eguagliano due slice della funzione total variance, riarrangiando e facendone il quadrato si ottiene una equazione polinomiale quadratica della forma

$$\mu_4 k^4 + \mu_3 k^3 + \mu_2 k^2 + \mu_1 k^1 + \mu_0 = 0;$$

Se questo polinomio quadratico non ha radici reali, allora le slice non si intersecano.

Nel caso in cui siano presenti intersezioni ($n > 0$), si definiscono i punti $\widetilde{k}_{\mathrm{i}}$ come

$$\widetilde{k}_1 = k_1 - 1,$$
$$\widetilde{k}_{\mathrm{i}} = \frac{1}{2}(k_{\mathrm{i}-1} + k_{\mathrm{i}}), \quad \mathrm{se} 2 \leq I \leq n,$$
$$\widetilde{k}_{\mathrm{n}+1} = k_{\mathrm{n}} + 1.$$

Per ognuno dei $n + 1$ punti $\widetilde{k}_{\mathrm{i}}$, si calcola la quantità c_{i} dove le due curve si intersecano:

$$c_{\mathrm{i}} = \max\left[0, w(\widetilde{k}_{\mathrm{i}}; \chi_1) - w(\widetilde{k}_{\mathrm{i}}; \chi_2)\right].$$

Definizione 4. *La Crossedness tra due slice della superficie SSVI è definita come il massimo $c_{\mathrm{i}}(i = 1, \ldots, n)$. Nel caso $n = 0$ la Crossedness è nulla.*

FUNZIONE SSVI

Questa funzione non fa nient'altro che ricevere in ingresso i parametri, le *Theta* **(volatilità ATM modificate tramite la formula** (5.1)**) e gli strike forward e restituire la varianza totale implicita** $w(k, t)$ **secondo la formula** (5.8)**.**

Algoritmo 12: GJ - SSVI

Function [w,phi] = `SSVI` (strike forward→ k, Theta, param→ x, FLAG)

% funzione che calcola la total implied variance w dati i parametri, gli strike e le vola ATM;
A seconda del valore di FLAG calcola w per uno strike o per tutta una slice specifica, oppure per la superficie intera;
eta ← primo elemento di x;
rho ← secondo elemento di x;
gamma ← terzo elemento di x;
phi ← funzione *Power-law parameterization* dell'equazione (5.13) dipendente da $Theta$,eta e $gamma$;

w $\leftarrow \frac{Theta}{2} * (1 + rho * phi * k + \sqrt{(phi * k + rho)^2(1 - rho^2)})$;

end;

Viene utilizzata sia per la generazione della prima φ e sia all'interno del ciclo-for: estrae i due valori η e ρ, calcola il valore della funzione φ secondo la formula della Power-law parameterization modificata (5.13) e compone la funzione $w(k,t)$ al variare del vettore degli strike k.

5.5 Risultati

Lo scopo principale del modello implementato è di creare la volatilità locale partendo dalla superficie di volatilità implicita generata tramite il modello SSVI. Per analizzare i risultati ottenuti anche in questo caso sono stati confrontati i prezzi calcolati dal modello con quelli di mercato, l'errore nel riprezzare un valore mancante e il tempo di esecuzione del codice.

Nella prima parte del codice viene fatta una calibrazione di tutti i parametri della superficie implicita, la quale poi viene modificata uno step alla volta, ma è possibile far vedere che le modifiche influenzano in piccola parte il risultato poiché, utilizzando i particolari dati scaricati, la funzione Crossedness non va ad influire particolarmente sulla scelta dell'ottimo locale trovato in precedenza[8]. Il fatto di scandire una scadenza alla volta è servito più che altro per la generazione della superficie di volatilità locale tramite la funzione (5.14). In Figura 5.2 è possibile vedere uno zoom della volatilità implicita calibrata dal codice in relazione ai valori scaricati dal mercato (quest'ultimi segnati dai

[8]Non per questo svolge un ruolo di secondo piano, anzi garantisce seppur muovendo di poco i parametri che venga evitata la possibilità di arbitraggio in tutta la superficie.

pallini rossi), mentre in Figura 5.3 è raffigurata la volatilità locale ottenuta alla fine del codice.

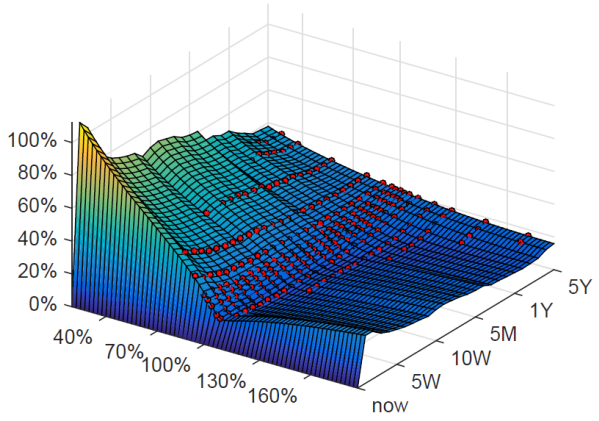


Figura 5.2: Volatilità implicita ottenuta con SSVI e volatilità di mercato (pallini rossi) al variare di T e di K/S_0.

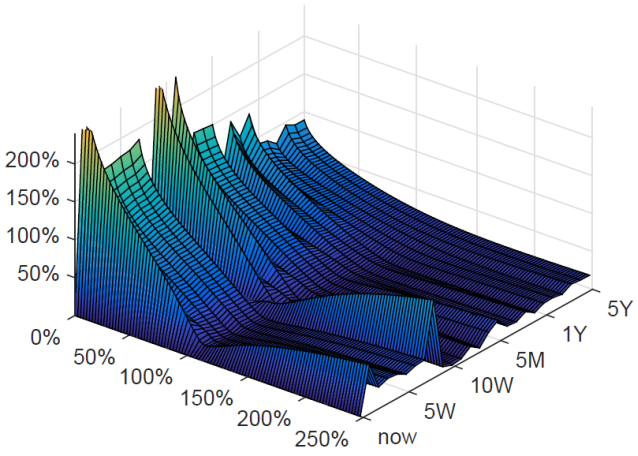


Figura 5.3: Superficie di volatilità locale generata dal modello G-J.

La volatilità implicita ottenuta dal modello SSVI interpola abbastanza bene quella scaricata dal mercato, confermandoci la correttezza del modello e la scelta di utilizzarlo come punto di partenza per passare poi alla lo-

cale[9]. Dalla rappresentazione in Figura 5.3 della superficie locale si vede che differisce notevolmente dalle superfici in Figura 4.5 e 4.9(a) ottenute con Andreasen-Huge e successive modifiche: la superficie è costante a tratti nel tempo, nel senso che cambia valore solo in concomitanza delle maturity della griglia (dal grafico si capisce poco perchè MATLAB cerca comunque di disegnare un'interpolazione tra i punti della superficie), ma negli strike segue un andamento curvilineo più che "a gradini", derivante dal fatto che l'implicita dalla quale deriva segue una curva parametrica ben precisa.

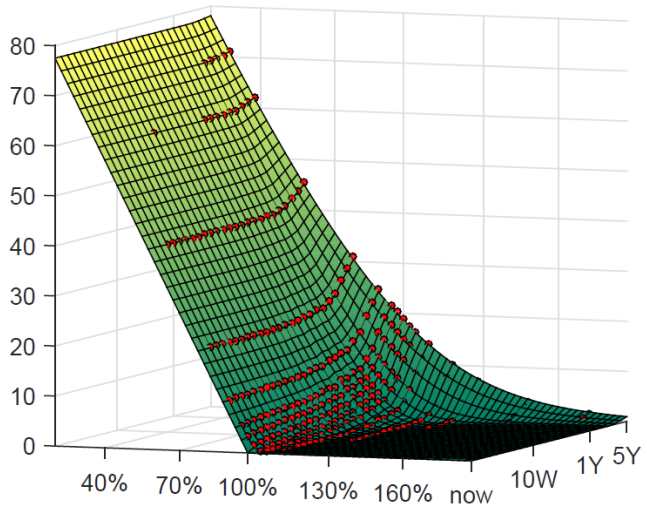


Figura 5.4: prezzi di mercato (pallini rossi) e prezzi ottenuti dal modello G-J, al variare del tempo e di K/S_0.

Utilizzando la volatilità locale è stato possibile calcolare il prezzo delle opzioni Call per ogni scadenza e strike. In Figura 5.4 è possibile osservare questi prezzi calcolati con Differenze Finite, al variare dei due parametri K e T nell'intorno dell'ATM, in relazione a quelli di mercato. I prezzi generati sono crescenti sia nella variabile tempo che nella variabile strike, garantendo quindi l'assenza di arbitraggio, anche se non interpolano benissimo quelli scaricati: si nota che c'è una sottostima dei prezzi ITM ed una sovrastima degli OTM, comprensibile dalla porzione dei pallini visibile nel grafico.

L'errore, presente in Tabella 5.1, è decisamente più grande rispetto al modello non parametrico, circa il doppio rispetto al *bid-ask spread*. Anche utilizzando il metodo Monte Carlo la situazione non migliora: RMSE=

[9]L'interpolazione delle volatilità scaricate non è perfetta, ma operando l'ottimizzazione step-by-step la situazione migliora ulteriormente, ricordando comunque che questa serve solo come punto di partenza per ottenere poi la superficie locale.

modello	Root Mean Squared Error	Leave-one-out	tempo
G-J	0.2101	0.1063	9.728 s

Tabella 5.1: Errore e tempo di esecuzione del modello Gatheral-Jacquier.

0.2282 e L-o-o= 0.3084, rendendo così evidente come il modello SSVI con volatilità locale non sia particolarmente adatto al semplice ricalcolo dei dati di mercato[10].

Per quanto riguarda il tempo di esecuzione invece, nel modello Gatheral-Jacquier si assiste ad un calo drastico dei secondi necessari alla calibrazione e creazione delle superfici dei prezzi e delle volatilità. Questo principalmente è spiegato dal fatto che in G-J il numero di parametri da stimare è sempre 2, mentre in A-H e A-H-K ad ogni istante vengono stimati un numero di variabili quante le volatilità non nulle presenti (mediamente una dozzina); inoltre confrontando direttamente le volatilità si evita il passaggio di dover calcolare, per ogni valore di volatilità provato dall'ottimizzazione, il prezzo dell'opzione necessario per il confronto, per ogni strike presente nella griglia.

[10]Questo si evince anche osservando la Figura 5.4: la superficie generata dal modello non interpola efficacemente i punti rappresentanti il mercato come invece fa il modello A-H e A-H-K.

Capitolo 6

Replica di opzioni path dependent

Come esposto in precedenza, la volatilità locale è stata pensata ed utilizzata per prezzare opzioni più complesse rispetto alle *plain vanilla*, è stato quindi necessario calcolare alcune di queste opzioni per poter capire quale tra i modelli implementati lavori meglio a questo scopo.

Le opzioni che si incontrano maggiormente nella pratica, oltre alle europee, appartengono alla famiglia delle *path dependent*. Tali opzioni sono caratterizzate dal fatto che il payoff a scadenza o all'esercizio dipenda in via non banale dalla storia passata del sottostante e dal valore spot alla scadenza o al momento dell'esercizio. In pratica ogni prodotto finanziario il cui valore dipende dal percorso e dai valori assunti dal sottostante per tutta la durata del contratto rientrano in questa categoria. La maggiorparte di queste opzioni non è standard e quotata sul mercato, ma sono commerciate *over the counter* cioè al di fuori dei circuiti borsistici ufficiali, dove non esiste un regolamento preciso; questo aumenta la necessità di avere i prezzi ti dali prodotti il più preciso possibile e relazionato al prezzo di opzioni scambiate nei mercati ufficiali.

Tra le varie opzioni path dependent si è scelto di prezzare le opzioni barriera, ovvero quelle dove il valore finale dipende dal raggiungimento, da parte del sottostante, di un livello denominato **barriera**. Esse possono cessare di esistere quando l'evento Barriera si verifica, oppure attivarsi in quell'istante, prendendo il nome *out* o *in* rispettivamente.

Oltre alle opzioni europee Call e Put, anche le Barriera vengono utilizzate come strumenti base per costruire derivati molto più strutturati; diventa quindi importante avere un modello che prezzi al meglio anch'esse, così da ottenere anche il prezzo dei prodotti strutturati il più accurato possibile.

Per calcolare il prezzo delle opzioni che dipendono dal percorso del sottostante non esistono quasi mai EDP esplicite, ciò fa sì che il metodo Monte Carlo sia l'unica strada percorribile per ottenere questi prezzi. E' quindi utilizzando Monte Carlo che sono state calcolate le opzioni barriera sia con il modello G-J sia con A-H-K per poter decretare quale tra i due sia meglio usare in questa circostanza. Va ricordato che il metodo A-H non viene preso in considerazione in questo capitolo perché era già stato confrontato con il metodo A-H-K risultando il meno accurato se accoppiato al metodo Monte Carlo.

Proposizione 6 (in-out parity [11])**.** *Il prezzo di un'opzione barriera, per esempio* in*, è uguale al prezzo di un'opzione europea con medesima scadenza e strike meno il prezzo di un'opzione* out *con medesima scadenza, strike e barriera.*

Se al tempo iniziale infatti si ha a disposizione una versione *down-and-in*[1] di una certa opzione Z con una barriera B e una versione *down-and-out* della stessa Z con la stessa barriera B, a scadenza equivale ad avere direttamente l'opzione Z.

L'utilizzo della in-out parity è fondamentale per poter valutare i prezzi delle barriere ottenuti dai due modelli, poiché è possibile grazie ad essa associarli ai prezzi scaricati dal mercato delle opzioni Call. Il procedimento consiste infatti nel prezzare separatamente una sola *out* e una sola *in* con medesima scadenza e strike, e valutare quanto la loro somma combaci o meno con il valore della Call europea corrispondente; così facendo si ha una sorta di riferimento con il mercato del valore che le barriere dovrebbero assumere e valutare quale modello lavori meglio nel calcolo di una opzione specifica.

Si è deciso di calcolare il prezzo delle due opzioni *in* e *out* entrambe con strike 102.5% e scadenza 5 anni e con livello di barriera $B = 95\%$, così da riferirsi allo stesso prezzo di mercato, ovvero $C_{\mathrm{K=102.5,T=5}} = 17.931$, utilizzato in precedenza nella tecnica Leave-one-out perché scelto nella zona dove gli errori dei modelli sono maggiori.

In Tabella 6.1 sono presenti i prezzi delle opzioni *down-and-in* e *down-and-out* calcolati separatamente, la loro somma così da ottenere il prezzo dell'opzione Call secondo la in-out parity e la differenza in valore assoluto tra esso ed il prezzo della stessa Call presente sul mercato.

Guardando la distanza dal prezzo scaricato, che rappresenta in ultima analisi la bontà con la quale i modelli prezzano le opzioni barriera, si conclude

[1] Si indica con *down* l'opzione la cui barriera è più bassa del valore iniziale S_0, e con *up* con barriera più alta di S_0.

modello	Down & In	Down & Out	Somma	Differenza col mercato
G-J	13.0591	4.6754	17.7345	0.1965
A-H-K	13.1247	4.6059	17.7306	0.1926

Tabella 6.1: Prezzo delle medesime opzioni barriera utilizzando i due modelli implementati.

che il modello non parametrico di Abasto, Hientzsch e Kust lavora meglio, seppur di poco, rispetto al parametrico di Gatheral e Jacquier; rendendolo così il modello "vincitore" del confronto.

Poichè i risultati dei due modelli non discostano significativamente, si è deciso di calcolare altre opzioni *down-and-out* al variare della scadenza e dello strike (ma mantenendo $B = 95\%$) così da vedere se ci sono zone della superficie dove i valori divergono. In Tabella 6.2 si nota che bene o male le differenze percentuali (A-H-K - G-J)/A-H-K sono contenute e non esistono zone con risultati differenti che potrebbero far pensare ad un errore sistematico.

K\T	1W	2M	5M	1Y	5Y
120%	–	–	0.09556	0.85155	0.03961
100%	0.14196	0.02499	0.00771	0.04107	0.010575
80%	0.010485	0.02102	0.03218	0.01159	0.00521

Tabella 6.2: Differenza percentuale tra i prezzi dell'opzione *down-and-out* ottenuti dai due modelli per diversi T e K.

Sempre per vedere quanto i modelli possano divergere o meno è stato fatto un confronto per quanto riguarda il prezzo di un opzione barriera ma con strike e scadenza al di fuori della griglia iniziale. Poichè entrambi gli algoritmi generano una superficie continua, come d'altronde è necessario per l'equazione di Dupire, si è voluto testare quanto coerentemente i due modelli prezzino un'opzione con scadenza $T = 2,5$ anni e strike $K = 98\%$ e solita barriera $B = 95\%$, senza un riferimento al mercato ma semplicemente valutando la distanza tra di essi.

Oltre alle opzioni barriera, è stato condotto anche un confronto che ha riguardato le opzioni asiatiche. Un'opzione path dependent è denominata *asiatica* se il suo valore dipende da una media nel tempo dei valori assunti dal titolo sottostante. Esistono diverse versioni di asiatica: in questa tesi si è scelto di prezzare l'opzione *average rate*, con media aritmetica (su tutti i

sotto intervalli) e sostituita a scadenza al posto del valore finale. Il payoff a scadenza assume quindi la forma $max\big[0, (\sum_{t=1}^{N_t} S_t)/N_t - K\big]$, ed è stato calcolato per $T = 5$ anni e $K = 102.5\%$.

modello	*down-and-out* fuori griglia	asiatica $(media - K)^+$
G-J	3.5419	2.3021
A-H-K	3.4521	2.3459

Tabella 6.3: Prezzo dell'opzione barriera con K e T non inizilmente presenti e della asiatica con media aritmetica utilizzando i due modelli studiati.

Dalla Tabella 6.3 riguardante le ultime due tipologie di prezzi da paragonare arriva un'ulteriore conferma del fatto che il modello A-H-K ed il G-J lavorano circa allo stesso modo. Se utilizzati insieme a Monte Carlo i valori che producono non si discostano molto sia per la "nuova" opzione barriera sia per l'opzione asiatica qui considerata.

Conclusioni

In questa tesi sono stati paragonati e valutati diversi modelli per l'utilizzo della volatilità locale tali da garantire l'assenza di arbitraggio. Per ogni modello è stato calcolato l'errore quadratico medio nel replicare i dati presi dal mercato, l'errore nel ricalcolare un valore mancante ed il prezzo di opzioni il cui valore dipende dal percorso del sottostante.

Analizzando i risultati ottenuti è stato possibile decretare il modello di Andreasen e Huge (senza modifiche e con Differenze Finite) quale modello da utilizzare per la generazione della superficie locale e dei prezzi europei, generando un errore molto più piccolo degli altri in un tempo accettabile. Per quanto riguarda il calcolo delle opzioni *path dependent*, particolarmente importanti visto che è l'area dove viene maggiormente utilizzata la volatilità locale, il miglior modello invece risulta essere quello di Abasto, Hientzsch e Kust con il metodo di Monte Carlo, avendo performato meglio del modello di Gatheral e Jacquier nel prezzare le barriera *in* e *out*.

In ultima analisi quindi si può dire che il modello A-H-K è il modello che risponde meglio alle esigenze, poiché produce una superficie locale meno accurata nel riprezzare i dati di mercato rispetto a A-H ma la utilizza meglio durante il calcolo dei derivati, scopo per la quale infatti viene creata. Volendo spezzare una lancia anche in favore del modello di G-J si può dire che produce una volatilità locale non particolarmente precisa, ma si ottengono comunque dei buoni risultati per il prezzo delle barriera e delle asiatiche, richiedendo però un minor costo computazionale rispetto agli altri modelli. Può essere quindi impiegato qualora si voglia ottenere un prezzo delle barriera indicativo, grezzo, ma in brevissimo tempo, caratteristica apprezzata in alcuni casi dai *practitioners*, soprattutto se si dispone già, per altri motivi, della superficie della volatilità implicita, così da ottenere immediatamente quella locale e quindi i prezzi.

Appendice A

Analisi dei dati scaricati

Per avere un confronto col mondo reale, si è scelto di valutare i risultati ottenuti in questa tesi alla luce dei prezzi delle opzioni utilizzate realmente dai trader sul mercato azionario. Ovviamente la discriminante più importante nella scelta delle opzioni è quale sottostante prendere in considerazione: e dovrà rispettare determinate caratteristiche, giustificate nei capitoli dove entreranno in gioco a seconda del modello considerato. La scelta è quindi ricaduta su uno tra i più rilevanti indici azionari, lo Standard and Poor's 500 Index, in quanto indice di riferimento dell'economia americana, la più importante al mondo.

In questo capitolo verrà anzitutto data qualche informazione generale su questo indice e del perché è stato scelto, poi si analizzeranno forma e contenuto dei dati presi dal mercato, e infine le modifiche che sono state apportate ad essi per poter essere utilizzati nei tre modelli confrontati più avanti.

A.1 L'indice S&P 500

Come spiegato in [43], lo Standard & Poor's 500 (indicato sulle piattaforme informatiche coi codici SPX o GSPC) è il più importante indice azionario nordamericano. Sebbene storicamente siano nati prima gli indici Dow Jones, questo paniere ha assunto maggiore importanza presso gli investitori. È infatti il principale riferimento relativo ai titoli quotati a Wall Street ed è uno tra i sottostanti più utilizzati per i prodotti derivati, quali futures, opzioni e certificates.

Questo paniere, creato da Standard & Poor's, una divisione di McGraw-Hill, viene calcolato dal 4 marzo 1957. La sua introduzione è stata possibile solo grazie alle funzionalità di calcolo avanzate e complesse possibili con i

progressi nel campo dell'elettronica. Prima del 1957, quando ancora non c'erano i computer, infatti, l'indice di S&P conteneva solamente 90 titoli. Ora esso contiene *500 titoli* azionari di grandi aziende quotate sul NYSE e sul NASDAQ con un'elevata liquidità (cioè sono sempre presenti molti ordini di acquisto e vendita su di essi, rendendo possibile in qualsiasi istante e per qualsiasi quantità la loro negoziazione) e vengono scelte in base alla loro capitalizzazione flottante[1]. Sebbene la maggior parte di questi titoli siano relativi ad aziende statunitensi, il criterio geografico non è una discriminante: attualmente sono incluse nell'indice 11 società estere.

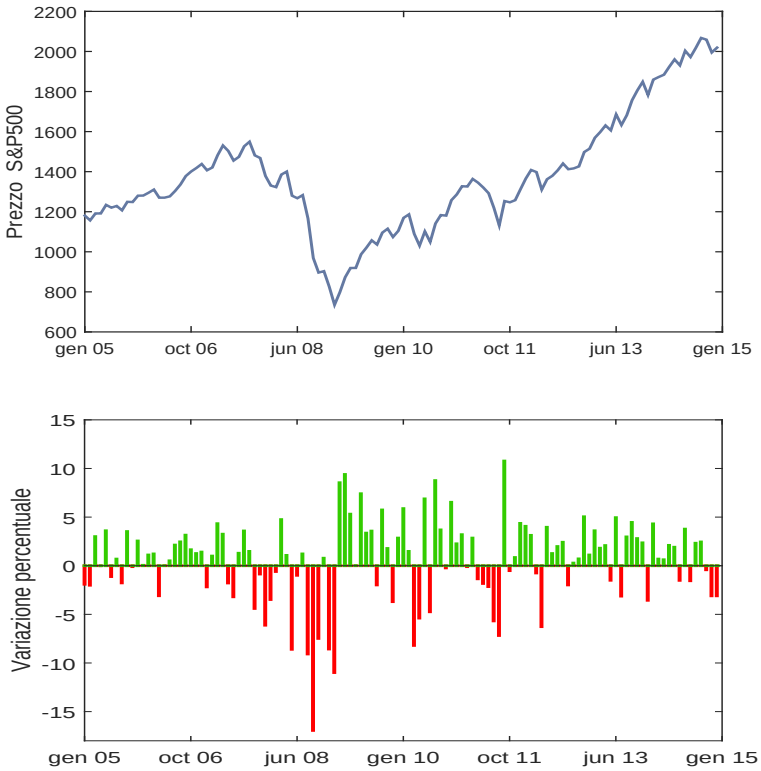


Figura A.1: Andamento dell'indice SP500 negli ultimi 10 anni.

Dal momento che è considerato il titolo che meglio rappresenta il mercato americano, ed essendo quest'ultimo il più grande e importante al mondo,

[1]Il flottante è la parte di capitale sociale che può essere comprata e venduta sul mercato, cioè titoli non detenuti dagli Stati, dai blocchi di controllo o da azionisti legati da patti di sindacato. La capitalizzazione flottante è semplicemente il prodotto del numero di titoli flottanti moltiplicati per i prezzi di detti titoli.

lo S&P viene spesso usato come benchmark per l'analisi di strategie di investimento, dove si intende *battere il mercato* quando la performance della strategia supera la performance dell'indice, nell'intervallo di tempo considerato. Alla luce di ciò è quindi ragionevole utilizzarlo come sottostante per confrontare le opzioni studiate, che siano plain vanilla o prodotti strutturati complessi. Si sarebbero potuti usare anche altri indici molto liquidi e ed altrettanto utilizzati come sottostante come l'EUROSTOXX50, il rispettivo europeo come importanza o, stringendo ancora di più il focus, il FTSEMIB italiano, ma alla data di raccolta dei dati (16/01/15) il mercato europeo era particolarmente volatile a causa dell'attesa dell'annuncio da parte della BCE sul piano del Quantitative Easing la settimana seguente, rendendo la volatilità implicita At-The-Money più alta nel breve periodo rispetto a quello lungo. Come verrà spiegato nel Capitolo 5, il modello Gatheral-Jacquier prevede, per avere assenza di arbitraggio, che si scelgano dati in condizioni di mercato standard, quindi con la struttura a termine della superficie costante o comunque non fortemente decrescente, quindi si è dovuto rivolgere la scelta verso il mercato americano (in quel momento più tranquillo) per poter soddisfare questa condizione senza dover modificare il modello.

A.2 I dati presi dal mercato

In linea con quanto esposto nel Capitolo 1, non tutti i prezzi delle opzioni vanilla sono direttamente consultabili sui principali sistemi di gestione dei dati finanziari quali Bloomberg o Reuters [44], ma vengono tabulate solo le volatilità implicite dei prezzi esistenti. La Tabella A.1 è un tipico output dove nella matrice sono presenti solo alcuni valori considerati principali, sia per quanto riguarda la scadenza, rappresentata in frazioni d'anno, sia per gli strike, espressi come percentuali del valore spot del sottostante, cioè il valore assunto al giorno dell'estrapolazione dei dati.

I dati presenti nella Tabella A.1 sono esattamente quelli utilizzati nella tesi: rappresentano tutti i valori di volatilità implicita di S&P500 estrapolata in data 16 gennaio 2015 presenti sul mercato. In quella data il valore dell'indice era uguale a 2019.2 USD, corrispondente alla riga con valore di strike 100%. Bloomberg stampa a video anche i valori forward del sottostante, cioè una stima del valore che può assumere nelle date future, perché in alcuni modelli, come tra l'altro il modello di Gatheral-Jacquier del Capitolo 5, viene utilizzata una rappresentazione diversa degli strike, collegata al valore forward F_{t} tramite l'equazione $k = \log\left(\frac{K}{F_{\mathrm{t}}}\right)$ chiamata moneyness.

La matrice in Tabella A.1 è formata da 17 percentuali dello strike e 26 scadenze differenti, ma molti valori non sono presenti, in quanto Bloomberg mostra solo le volatilità implicite che si riferiscono a coppie (k, t) davvero usate dai trader. Si può notare infatti che al passare del tempo, valori molto lontani dal prezzo spot diventano più probabili, e quindi *tradabili*, mentre nel breve periodo, una settimana o un mese, difficilmente il prezzo diventerà la metà o raddoppierà. Questo però in determinate situazioni può avverarsi, ed estrarre la superficie di volatilità diventa quindi l'unico modo per ottenere questi valori mancanti, oltre a quelli con strike e maturity non presenti nella griglia della matrice, necessari per prezzare particolari strumenti e con particolari scadenze.

A.2.1 Modifiche apportate

I dati rappresentati in questo modo però non sono di facile gestione se lo scopo non è la mera consultazione, quindi si è dovuto modificarli per poterli utilizzare in modo conveniente nei modelli implementati nei capitoli successivi.

Le modifiche hanno interessato:

Gli strike Si può notare che le percentuali non sono equidistanti ma si infittiscono intorno all'*at the money* (100%) con l'intervallo più piccolo largo 2,5%. Per ragioni di implementazione però risultava più conveniente avere una griglia equidistante negli strike quindi è stata creata una matrice più larga aggiungendo righe dove serviva in modo tale da avere un intervallo sempre uguale a 2,5%. Come è stato spiegato nel capitolo 4, a causa delle condizioni al bordo (4.4) sono state aggiunte righe sopra il 175% e sotto il 30%, arrivando a coprire un intervallo da 0% a 250%. La matrice in Tabella A.1 è così diventata di dimensione 101×26.

Le scadenze Anche le scadenze non sono equidistanti, gli intervalli si allargano al passare del tempo, e coprono uno spazio temporale che va da una settimana fino a 5 anni, come mostrato in Figura 4.1(a). Questo però non è stato un problema nella stesura dei codici quindi in un primo momento si è lavorato con la matrice con 26 scadenze, per poi infittire la griglia di un fattore 10 all'interno di ogni codice per avere un'accuratezza migliore del risultato.

Gli spazi vuoti Per poter gestire i punti della matrice senza valore, la scelta più immediata e semplice è stata quella di lasciare i vuoti (o meglio con-

siderarli come volatilità dello 0%) e calcolare le stime basandosi esclusivamente sui pochi valori presenti, sia per quanto riguarda il confronto tra volatilità, sia tra i prezzi generati da esse.

Un altro fatto da mettere sotto i riflettori è che sono presenti maggiormente valori con strike *out of the money* ($< 100\%$), rispetto agli *in the money*($> 100\%$): questo è dovuto al fatto che nel mercato equity, dove si trova anche il sottostante scelto, la superficie presenta poco Smile e molta Skew sui valori negativi[2], in quanto il mercato è preoccupato maggiormente dal fatto che un titolo perda valore, o peggio crolli, rispetto ad un'impennata improvvisa, e questa ansia e incertezza si tramuta in una volatilità più alta e accurata (il tutto ha in effetti un fondamento di verità in quanto storicamente, e soprattuto dopo la crisi petrolifera del 1973, nel mercato dei titoli azionari si sono registrate molte cadute improvvise a fronte di crescite lente e di lungo periodo). Lo *scopo del gioco* quindi consiste nel riempire gli spazi vuoti ed ottenere una superficie vera e propria, nel miglior modo possibile.

Grazie alla biunivocità tra volatilità implicita e prezzi, è possibile passare in ingresso ai codici MATLAB esattamente questa matrice e poi far calcolare al computer il prezzo dell'opzione vanilla Call[3]; questo perché in alcuni casi come nel modello Andreasen-Huge del Capitolo 4 e Abasto-Hientzsch-Kust della sezione 4.5 l'ottimizzazione è stata fatta lavorando sui possibili valori della volatilità, mentre per il modello Gatheral-Jacquier del Capitolo 5 avendo già a disposizione una forma parametrica si sono scelti i parametri in modo tale da minimizzare la distanza tra i prezzi generati e quelli di mercato.

[2]Si rimanda al capitolo 1 per il significato finanziario di Smile e Skew

[3]La scelta di prezzare una Call piuttosto che una Put è irrilevante grazie alla Put-Call Parity.

	16-Jan-15	23-Jan-15	30-Jan-15	6-Feb-15	13-Feb-15	20-Feb-15	27-Feb-15	6-Mar-15	13-Mar-15
days	0	7	14	21	28	35	42	49	56
fraction	0.000	0.019	0.038	0.058	0.077	0.096	0.115	0.134	0.153
indicative	now	1W	2W	3W	1M	5W	6W	7W	2M
forward	2019.16	2019.07	2018.78	2017.42	2016.14	2015.09	2014.56	2013.65	2012.76
176.0%									
175.0%									
150.0%									
130.0%									
120.0%									
110.0%			14.32%	14.83%	13.77%	12.69%	12.53%	12.35%	12.18%
105.0%		14.10%	14.77%	14.33%	13.51%	12.64%	12.82%	12.90%	13.03%
102.5%		16.26%	16.59%	16.16%	15.27%	14.42%	14.65%	14.72%	14.85%
100.0%		20.19%	19.88%	19.38%	18.16%	17.07%	17.18%	17.13%	17.12%
97.5%		23.44%	23.10%	22.26%	20.84%	19.52%	19.53%	19.39%	19.24%
95.0%		26.04%	25.71%	24.99%	23.37%	21.76%	21.68%	21.45%	21.20%
90.0%		35.69%	32.29%	30.67%	28.60%	26.54%	26.14%	25.62%	25.19%
80.0%			44.38%	42.56%	39.46%	36.56%	35.35%	34.11%	33.22%
60.0%				49.00%	45.31%	43.66%	42.40%	41.74%	41.48%
40.0%									
30.0%									
29.0%									

Tabella A.1: Volatilità implicite dell'indice SPX scaricate da Bloomberg in data 16-gen-15, in funzione degli strike e delle scadenze. Sono presenti oltre alle date delle scadenze anche i giorni corrispondenti, il riferimento in mesi/anni e la frazione d'anno. Bloomberg restituisce anche i valori Forward dell'indice ATM. *continua...*

	20-Mar-15	27-Mar-15	2-Apr-15	17-Apr-15	30-Apr-15	29-May-15	19-Jun-15	30-Jun-15	18-Sep-15
days	63	70	76	91	104	133	154	165	245
fraction	0.173	0.192	0.208	0.249	0.285	0.364	0.422	0.452	0.671
indicative	9W	10W	11W	3M	3,5M	4,5M	5M	5,5M	8M
forward	2012.69	2012.35	2011.99	2011.20	2010.86	2007.20	2005.39	2005.00	1998.27
176.0%									
175.0%									
150.0%									
130.0%								11.67%	
120.0%	12.63%	12.78%	12.85%	12.50%	12.14%	11.62%	11.45%	11.40%	11.79%
110.0%	11.89%	12.12%	12.08%	11.97%	12.11%	12.35%	12.57%	12.76%	13.60%
105.0%	13.19%	13.41%	13.57%	13.73%	14.04%	14.38%	14.64%	14.75%	15.45%
102.5%	15.02%	15.18%	15.26%	15.35%	15.59%	15.77%	15.96%	16.01%	16.50%
100.0%	17.11%	17.20%	17.20%	17.15%	17.27%	17.23%	17.33%	17.33%	17.58%
97.5%	19.03%	19.10%	19.05%	18.87%	18.90%	18.68%	18.68%	18.66%	18.65%
95.0%	20.85%	20.88%	20.76%	20.45%	20.45%	20.10%	19.99%	19.95%	19.73%
90.0%	24.79%	24.51%	24.17%	23.63%	23.52%	22.98%	22.64%	22.54%	21.93%
80.0%	32.64%	31.79%	31.33%	30.40%	29.81%	28.63%	28.02%	27.75%	26.28%
60.0%	42.01%	40.73%	42.08%	41.65%	40.98%	39.14%	38.57%	38.25%	35.05%
40.0%	49.35%								40.75%
30.0%									
29.0%									

Tabella A.2: Volatilità implicite dell'indice SPX, continuazione della tabella precedente per scadenze maggiori.
continua...

	30-Sep-15	18-Dec-15	15-Jan-16	17-Jun-16	31-Dec-16	31-Dec-17	31-Dec-18	31-Dec-19
days	257	336	364	518	715	1080	1445	1810
fraction	0.704	0.921	0.997	1.419	1.959	2.959	3.959	4.959
indicative	8,5M	11M	1Y	1,5Y	2Y	3Y	4Y	5Y
forward	1997.96	1991.61	1990.63	1981.15	1975.30	1969.05	1969.64	1974.35
176.0%								
175.0%							15.69%	16.27%
150.0%					12.65%	14.58%	15.67%	16.70%
130.0%		11.60%	11.13%	12.36%	13.63%	15.68%	16.96%	17.98%
120.0%	11.92%	12.18%	12.26%	13.61%	15.02%	16.91%	18.03%	18.93%
110.0%	13.74%	14.38%	14.51%	15.78%	16.94%	18.49%	19.34%	20.07%
105.0%	15.53%	16.05%	16.09%	17.11%	18.04%	19.39%	20.09%	20.71%
102.5%	16.55%	16.95%	16.96%	17.80%	18.61%	19.86%	20.48%	21.05%
100.0%	17.61%	17.85%	17.85%	18.50%	19.19%	20.34%	20.88%	21.40%
97.5%	18.67%	18.76%	18.75%	19.21%	19.78%	20.83%	21.29%	21.76%
95.0%	19.73%	19.68%	19.63%	19.93%	20.38%	21.33%	21.71%	22.13%
90.0%	21.88%	21.53%	21.37%	21.39%	21.61%	22.34%	22.58%	22.90%
80.0%	26.09%	25.22%	24.87%	24.37%	24.14%	24.34%	24.38%	24.52%
60.0%	34.71%	32.94%	32.15%	30.26%	29.07%	28.29%	28.06%	28.02%
40.0%	40.93%	39.46%	40.05%	37.49%	35.12%	33.02%	32.51%	32.19%
30.0%				40.86%	38.48%	35.37%	35.18%	34.88%
29.0%								

Tabella A.3: Volatilità implicite dell'indice SPX, continuazione della tabella precedente per scadenze maggiori.

Appendice B

Il metodo Monte Carlo

Il Metodo Monte Carlo fa parte della famiglia dei metodi statistici non parametrici. Viene utilizzato per trarre stime attraverso simulazioni e si basa su un algoritmo che genera una serie di numeri casuali tra loro incorrelati, che seguono una certa distribuzione di probabilità in qualche modo collegata con il fenomeno da analizzare. Le sue origini risalgono alla metà degli anni 40 nell'ambito del Progetto Manhattan e il nome Monte Carlo fu inventato in seguito da Nicholas Constantine Metropolis in riferimento alla tradizione nei giochi d'azzardo dello stato omonimo nel sud della Francia, sede del più famoso casinò e quindi simbolo dell'aleatorietà.

Il metodo si basa sulla creazione di un numero molto elevato di possibili scenari, e dell'utilizzo della **legge dei grandi numeri**: La legge dei grandi numeri, detta anche legge empirica del caso oppure teorema di Bernoulli (in quanto la sua prima formulazione è dovuta a Jakob Bernoulli), descrive il comportamento della media di una sequenza di n variabili casuali indipendenti e caratterizzate dalla stessa distribuzione di probabilità (n misure della stessa grandezza, n lanci della stessa moneta ecc.) al tendere ad infinito della numerosità della sequenza stessa. In altre parole, grazie alla legge dei grandi numeri, *possiamo fidarci* che la media che calcoliamo a partire da un numero sufficiente di campioni sia *sufficientemente vicina* alla media vera.

Supponiamo per esempio di voler determinare il valore atteso μ di un modello statistico. Si esegua una simulazione, generando la variabile casuale X_1 in modo che μ sia il valore atteso di X_1. Consideriamo una seconda simulazione, generando una variabile casuale X_2 tale che il suo valore atteso sia sempre μ. Proseguiamo con n simulazioni, generando fino a n variabili casuali X_{n} indipendenti e identicamente distribuite con $\mathbb{E}[X_{\mathrm{n}}] = \mu$. A questo punto per stimare il valore atteso si utilizza la media aritmetica $\bar{X}$ (stimatore

non distorto) delle n variabili casuali generate, cioè

$$\bar{X} = \frac{\sum_{i=0}^{\infty} X_{\mathrm{i}}}{n} \tag{B.1}$$

con valore atteso $\mathbb{E}[\bar{X}] = \mu$ e varianza $var(\bar{X}) = \sigma^2/n$. La legge (forte) dei grandi numeri afferma che

$$P\Big(\lim_{n\to+\infty} \bar{X} = \mu\Big) = 1, \tag{B.2}$$

ossia la media campionaria converge *quasi certamente*[1] alla media comune delle X_{i}.

Tale tecnica è stata proposta per la prima volta nel campo finanziario in [8] dove sono state utilizzate le simulazioni per individuare dei possibili percorsi per il prezzo del bene sottostante e sfruttando l'assunzione di operare in un mercato privo di rischio per poter ricavare il valore dell'opzione come media attualizzata dei payoff a scadenza. Ricordando che il prezzo di un'opzione europea Call si ha da

$$C_{\mathrm{t}} = e^{-r(T-t)}\mathbb{E}\big[\max(S_{\mathrm{T}} - K, 0)\big] \tag{B.3}$$

è possibile utilizzare il metodo Monte Carlo per il calcolo del valore atteso:

$$\mathbb{E}\big[\max(S_{\mathrm{T}} - K, 0)\big] = \frac{1}{M}\sum_{i=0}^{M} \max(S_{\mathrm{i,T}} - K, 0). \tag{B.4}$$

Le simulazioni dei percorsi invece, che portano ad ottenere i valori a scadenza $S_{\mathrm{i,T}}$ variano a seconda delle ipotesi fatte circa l'andamento del sottostante, come evolve nel tempo, e quindi da quale distribuzione di probabilità estrarre le variabili casuali per differenziare ogni simulazione. Collegato a questo scopo e molto utile nella pratica è il **teorema centrale del limite**: dati X_{i} variabili aleatorie iindipendenti e identicamente distribuite con media $\mathbb{E}[X_{\mathrm{i}}] = \mu$ e varianza $var(X_{\mathrm{i}}) = \sigma^2$ allora

$$\frac{\sum_{i=1}^{n} X_{\mathrm{i}} - n\mu}{\sigma\sqrt{n}} \xrightarrow{d} N(0,1) \tag{B.5}$$

cioè all'aumentare delle traiettorie M il valore atteso si distribuisce come una normale con media μ e varianza σ^2/M. Si può quindi notare che la varianza

[1] Un modo più rigoroso per definire la convergenza quasi certa (o quasi ovunque) è dire che una successione f_{n} converge q.o. a una funzione f se l'insieme dei punti dove il suo limite per $n \to \infty$ non coincide con f ha misura nulla [28].

è inversamente proporzionale al numero delle simulazioni fatte e l'ordine di convergenza è del tipo $1/\sqrt{M}$.

Per quanto riguarda la simulazione del sottostante bisogna fare una precisazione: nella teoria di Black&Scholes (e con ottima approssimazione anche nella realtà) si ipotizza l'andamento del sottostante come un processo continuo, nel quale si hanno infiniti punti di discontinuità della derivata prima ma mai un punto di salto oppure istanti temporali dove non esistono valori. Nella pratica però ovviamente si deve farne un'approssimazione discreta del processo continuo, definendo una griglia temporale formata da n punti e quindi l'intervallo $\triangle t = T/N$ col quale campionare (oppure ricreare) l'andamento del sottostante.

E' possibile quindi procedere da un istante temporale al successivo conoscendo la probabilità di transizione da uno stato all'altro, oppure qualora si conoscesse la dinamica esatta discretizzare la soluzione, come si è fatto in questa tesi, supponendo che il sottostante evolvesse come un Moto Browniano Geometrico

$$dS_{\mathrm{t}} = rS_{\mathrm{t}}dt + \sigma S_{\mathrm{t}}dW_{\mathrm{t}}$$

oppure con schemi di approssimazione tipo quello di Eulero o di Milstein

$$X_{\mathrm{t_{i+1}}} = X_{\mathrm{t_i}} + \int_{t_{\mathrm{i}}}^{t_{\mathrm{i+1}}} \mu(X_{\mathrm{t}})dt + \int_{t_{\mathrm{i}}}^{t_{\mathrm{i+1}}} \sigma(X_{\mathrm{t}})dW_{\mathrm{t}}$$

$$Eulero: \; X_{\mathrm{t_{i+1}}} = X_{\mathrm{t_i}} + \mu(X_{\mathrm{t_i}})\triangle t + \sigma(X_{\mathrm{t_i}})\sqrt{\triangle t}Z$$

$$Milstein: \; X_{\mathrm{t_{i+1}}} = X_{\mathrm{t_i}} + \mu(X_{\mathrm{t_i}})\triangle t + \sigma(X_{\mathrm{t_i}})\sqrt{\triangle t}Z + \frac{1}{2}\sigma(X_{\mathrm{t_i}})^{\top}\sigma(X_{\mathrm{t_i}})\triangle t(Z^2-1)$$

molto flessibili ma con bias più grandi, utilizzabili sostanzialmente quando non si conosce la dinamica esatta del processo.

Si possono fare ulteriori considerazioni sul metodo Monte Carlo ma variano a seconda della scelta fatta per l'andamento del sottostante dell'opzione, quindi si rimanda al Capitolo 4.5 per la spiegazione della scelta fatta, per le considerazioni che ne derivano.

variabili antitetiche

E' necessario un piccolo excursus su un possibilie miglioramento del metodo base di Monte Carlo. Le simulazioni delle variabili indipendenti e identicamente distribuite Z estratte dalla normale standard sono un punto cruciale del metodo in esame, poiché sono l'unica fonte di aleatorietà del modello e devono essere il più accurate possibili, per ottenere risultati soddisfacenti.

Per migliorare le stime quindi sono state ideate diverse tecniche di **riduzione della varianza**, così da massimizzare il più possibile l'accuratezza utilizzando lo stesso numero di campioni: una tra le tecniche più impiegate con questo scopo è l'utilizzo delle variabili antitetiche.

La distribuzione normale standard è simmetrica, quindi se ϕ è la funzione densità di probabilità della normale standard vuol dire che $\phi(x) = \phi(-x)$; perciò quando viene simulata una Z_i è ugualmente probabile ottenere un certo valore X_i e $-X_\mathrm{i}$. Per essere sicuri che questa proprietà venga mantenuta durante le simulazioni, ogni volta che viene generato il valore X_i, il valore $-X_\mathrm{i}$ (il suo antitetico) è utilizzato in un altra simulazione della traiettoria, ottenendo il doppio del numero di campionamenti fatti dalla distribuzione.

Sia Y_i il valore di un opzione calcolato utilizzando un certo campione di andamento del sottostante, e sia $\widetilde{Y}_\mathrm{i}$ il valore calcolato con l'andamento antitetico. Y_i e $\widetilde{Y}_\mathrm{i}$ non sono indipendenti tra loro ma ogni coppia $(Y_\mathrm{i}, \widetilde{Y}_\mathrm{i})$ è indipendente e identicamente distribuita. Questo fa sì che la media può essere definita come

$$\bar{Y} = \frac{1}{2n}\left(\sum_{i=1}^{n} Y_\mathrm{i} + \sum_{i=1}^{n} \widetilde{Y}_\mathrm{i}\right) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{Y_\mathrm{i} + \widetilde{Y}_\mathrm{i}}{2}\right) \tag{B.6}$$

dove n è il numero di campioni originariamente estratti e $\frac{Y_\mathrm{i}+\widetilde{Y}_\mathrm{i}}{2}$ sono iid. Per la legge dei grandi numeri quindi si ha

$$\bar{Y} \to \mathbb{E}[Y] \quad per\ n \to \infty$$

e per il teorema centrale del limite

$$\frac{\bar{Y} - \mathbb{E}[Y]}{\sigma/\sqrt{n}} \to N(0,1) \quad per\ n \to \infty$$
$$\sigma^2 = VAR\big(\bar{Y}\big).$$

La riduzione della varianza utilizzando le variabili antitetiche è data quindi dal fatto che Y_i e $\widetilde{Y}_\mathrm{i}$ sono negativamente correlate, da cui segue che la varianza di $\bar{Y}$ è minore che se avessimo considerato $2n$ campioni di Y_i [23].

Appendice C

Il metodo delle differenze finite

Il metodo delle differenze finite consiste nell'approssimare il valore della derivata di una funzione u in un punto $\tilde{x}$ (per il quale sarebbe necessario conoscere tutti i valori della funzione, quindi infiniti, in un intorno di $\tilde{x}$), con un'espressione che ne tenga in conto solo un numero finito. Ciò permette ad esempio di trasformare un'equazione alle derivate parziali in un problema algebrico: in particolare si ottiene un sistema lineare del tipo $Ax = b$, con A matrice sparsa, la cui sparsità dipende dal numero di valori usati nell'approssimazione delle derivate [40].

Il più classico approccio per determinare le approssimazioni ed analizzarne l'errore, consiste nello sviluppare la funzione $u(x)$ in **serie di Taylor** in un intorno del punto $\tilde{x}$. Dato $h > 0$ si hanno:

$$u(\tilde{x}+h) = u(\tilde{x}) + hu'(\tilde{x}) + \frac{1}{2}h^2u^{(2)}(\tilde{x}) + \frac{1}{6}h^3u^{(3)}(\tilde{x}) + O(h^4), \tag{C.1}$$

$$u(\tilde{x}-h) = u(\tilde{x}) - hu'(\tilde{x}) + \frac{1}{2}h^2u^{(2)}(\tilde{x}) - \frac{1}{6}h^3u^{(3)}(\tilde{x}) + O(h^4), \tag{C.2}$$

Dalle due precedenti espressioni si ricavano le due seguenti approssimazioni del primo ordine per la derivata prima e l'approssimazione per la derivata seconda:

$$D_+u(\tilde{x}) = \frac{u(\tilde{x}+h) - u(\tilde{x})}{h} = u'(\tilde{x}) + O(h) \tag{C.3}$$

$$D_-u(\tilde{x}) = \frac{u(\tilde{x}) - u(\tilde{x}-h)}{h} = u'(\tilde{x}) + O(h) \tag{C.4}$$

$$D^2u(\tilde{x}) = \frac{1}{h^2}[u(\tilde{x}+h) - 2u(\tilde{x}) + u(\tilde{x}-h)] \tag{C.5}$$

Per risolvere l'equazione non resta altro che discretizzare i domini delle

variabili (rispetto alle quali bisogna fare la derivata) in un numero sufficientemente alto di intervalli, sostituire nell'equazione le corrispondenti approssimazioni, e risolvere il sistema di equazioni algebriche per il vettore di incognite. L'approssimazione (C.3) è chiamata *esplicita* (o Eulero in avanti) mentre la (C.4) è chiamata *implicito* (o Eulero all'indietro); Il metodo di Crank-Nicolson invece utilizza una combinazione lineare equipesata dei due precedenti [38]. Si riporta come esempio significativo l'equazione del calore:

$$\begin{cases} U_{\mathrm{t}} = U_{\mathrm{xx}} \\ U(0,t) = U(1,t) = 0, \quad U(x,0) = U_0(x). \end{cases} \tag{C.6}$$

Viene diviso lo spazio in $x_0, \ldots, x_{\mathrm{J}}$ e il tempo in $t_0, \ldots, t_{\mathrm{N}}$, chiamando h l'intervallo dei punti spaziali, r quello dei punti temporali e l'approssimazione della funzione nel punto j, n come $u_{\mathrm{j}}^n = u(x_{\mathrm{j}}, t_{\mathrm{n}})$. Usando l'approssimazione (C.3), l'equazione (C.6) diventa

$$\frac{u_{\mathrm{j}}^{n+1} - u_{\mathrm{j}}^n}{r} = \frac{u_{\mathrm{j+1}}^n - 2u_{\mathrm{j}}^n + u_{\mathrm{j-1}}^n}{h^2} \tag{C.7}$$

Supponendo di conoscere la soluzione all'istante n, si può ottenere l'istante successivo riscrivendo l'equazione

$$u_{\mathrm{j}}^{n+1} = \lambda u_{\mathrm{j-1}}^n + (1-2\lambda)u_{\mathrm{j}}^n + \lambda u_{\mathrm{j+1}}^n \qquad \lambda = \frac{r}{h^2}. \tag{C.8}$$

Facendo questo procedimento per tutti gli istanti temporali (noto il primo o l'ultimo a seconda delle condizioni al bordo) si risolvono insieme tutte le equazioni grazie alla forma matriciale, ottenendo così l'equazione $Ax = b$ dove x è il vettore contenente i valori della funzione conosciuti, A la matrice contenente le caratteristiche dell'equazione iniziale e le condizioni al bordo e b il vettore dei valori da determinare.

Si può dimostrare [13] che il metodo esplicito è numericamente stabile per $\lambda \leq 1/2$, mentre quello implicito lo è incondizionatamente[1]. Entrambi hanno un errore numerico con proporzione lineare nello step del tempo e quadratica in quello dello spazio:

$$\triangle u = O(r) + O(h^2). \tag{C.9}$$

mentre in Crank-Nicolson, oltre ad essere un metodo incondizionatamente stabile, la proporzione è quadratica in entrambi gli step, quindi fornisce a parità di finezza della griglia stime più accurate. La scelta di non utilizzare sempre Crank-Nicolson derivata dal fatto che può esibire *qualità indesiderabili*, come oscillazioni persistenti o valori inusuali, se la grandezza dello step nella dimensione del tempo è molto larga [39].

[1]Un metodo è incondizionatamente stabile se la stabilità della soluzione non dipende dalla dimensione degli intervalli considerati della griglia sottostante

Appendice D

Derivazione dell'equazione che lega la volatilità implicita alla locale

I prezzi delle opzioni presenti sul mercato sono quotati in termini della volatilità implicita di Black&Scholes $\sigma_{\mathrm{BS}}(K, T, S_0)$. In altre parole si può scrivere

$$C(S_0, K, T) = C_{\mathrm{BS}}(S_0, K, \sigma_{\mathrm{BS}}(K, T, S_0), T).$$

Tuttavia in questa sezione è più conveniente lavorare utilizzando solo due variabili adimensionali: la varianza implicita totale w definita come $w(k,t) = \sigma^2_{\mathrm{BS}}(k,t) \cdot t$ e la trasformazione *moneyness* definita come $k = \log\left(\frac{K}{F_{\mathrm{T}}}\right)$, dove $F_{\mathrm{T}} = S_0 \cdot exp\{\int_0^T r(t)\,dt\}$ è il prezzo forward del sottostante al tempo zero.

Utilizzando queste variabili la formula di Black&Scholes del prezzo futuro di un opzione diventa

$$\begin{aligned} C_{\mathrm{BS}}(F_{\mathrm{T}}, k, w) &= F_{\mathrm{T}}\{N(d_1) - e^k N(d_2)\} \\ &= F_{\mathrm{T}}\Big\{N\Big(-\frac{k}{\sqrt{w}} + \frac{\sqrt{w}}{2}\Big) - e^k N\Big(-\frac{k}{\sqrt{w}} - \frac{\sqrt{w}}{2}\Big)\Big\} \end{aligned} \tag{D.1}$$

e l'equazione di Dupire (2.2) diventa

$$\frac{\partial C}{\partial T} = \frac{\sigma^2_{\mathrm{L}}}{2}\Big\{\frac{\partial^2 C}{\partial k^2} - \frac{\partial C}{\partial k}\Big\} + r(t)C \tag{D.2}$$

con σ^2_{L} varianza locale. A questo punto calcolando le derivate della formula di B&S si ha

$$\frac{\partial^2 C_{\mathrm{BS}}}{\partial w^2} = \Big(-\frac{1}{8} - \frac{1}{2w} + \frac{k^2}{2w^2}\Big)\frac{\partial C_{\mathrm{BS}}}{\partial w}$$

$$\frac{\partial^2 C_{\mathrm{BS}}}{\partial w \partial k} = \Big(\frac{1}{2} - \frac{k}{w}\Big)\frac{\partial C_{\mathrm{BS}}}{\partial w}$$
$$\frac{\partial^2 C_{\mathrm{BS}}}{\partial^2 k} - \frac{\partial C_{\mathrm{BS}}}{\partial k} = 2\frac{\partial C_{\mathrm{BS}}}{\partial w}. \tag{D.3}$$

Possiamo trasformazione l'equazione (D.2) in un'equazione espressa in termini della varianza implicita eseguendo le sostituzioni

$$\frac{\partial C}{\partial k} = \frac{\partial C_{\mathrm{BS}}}{\partial k} + \frac{\partial C_{\mathrm{BS}}}{\partial w}\frac{\partial w}{\partial k}$$
$$\frac{\partial^2 C}{\partial k^2} = \frac{\partial^2 C_{\mathrm{BS}}}{\partial k^2} + 2\frac{\partial^2 C_{\mathrm{BS}}}{\partial w \partial k}\frac{\partial w}{\partial k} + \frac{\partial^2 C_{\mathrm{BS}}}{\partial w^2}\Big(\frac{\partial w}{\partial k}\Big)^2 + \frac{\partial C_{\mathrm{BS}}}{\partial w}\frac{\partial^2 w}{\partial k^2}$$
$$\frac{\partial C}{\partial T} = \frac{\partial C_{\mathrm{BS}}}{\partial T} + \frac{\partial C_{\mathrm{BS}}}{\partial w}\frac{\partial w}{\partial T} = \frac{\partial C_{\mathrm{BS}}}{\partial w}\frac{\partial w}{\partial T} + r(T)C_{\mathrm{BS}}$$

dove l'ultima equazione segue dal fatto che l'unica dipendenza esplicita dal T nel prezzo dell'opzione (D.1) è attraverso il prezzo forward $F_{\mathrm{T}} = S_0 exp\{\int_0^T r(t)\,dt\}$. L'equazione di Dupire (2.2) a questo punto diventa (considerando il tasso di interesse $r(t) = 0$)

$$\frac{\partial C_{\mathrm{BS}}}{\partial w}\frac{\partial w}{\partial T}$$
$$= \frac{\sigma_{\mathrm{L}}^2}{2}\Big\{ - \frac{\partial C_{\mathrm{BS}}}{\partial k} + \frac{\partial^2 C_{\mathrm{BS}}}{\partial k^2} - \frac{\partial C_{\mathrm{BS}}}{\partial w}\frac{\partial w}{\partial k} + 2\frac{\partial^2 C_{\mathrm{BS}}}{\partial k \partial w}\frac{\partial w}{\partial k}$$
$$+ \frac{\partial^2 C_{\mathrm{BS}}}{\partial w^2}\Big(\frac{\partial w}{\partial k}\Big)^2 + \frac{\partial C_{\mathrm{BS}}}{\partial w}\frac{\partial^2 w}{\partial k^2}\Big\}$$
$$= \frac{\sigma_{\mathrm{L}}^2}{2}\frac{\partial C_{\mathrm{BS}}}{\partial w}\Big\{2 - \frac{\partial w}{\partial k} + 2\Big(\frac{1}{2} - \frac{k}{w}\Big)\frac{\partial w}{\partial k}$$
$$+ \Big(- \frac{1}{8} - \frac{1}{2w} + \frac{k^2}{2w^2}\Big)\Big(\frac{\partial w}{\partial k}\Big)^2 + \frac{\partial^2 w}{\partial k^2}\Big\}.$$

Eliminando il fattore $\frac{\partial C_{\mathrm{BS}}}{\partial w}$ e semplificando, si ottiene

$$\frac{\partial w}{\partial T} = \sigma_{\mathrm{L}}^2\Big\{1 - \frac{k}{w}\frac{\partial w}{\partial k} + \frac{1}{4}\Big(- \frac{1}{4} - \frac{1}{w} + \frac{k^2}{w^2}\Big)\Big(\frac{\partial w}{\partial k}\Big)^2 + \frac{1}{2}\frac{\partial^2 w}{\partial k^2}\Big\}$$

ed invertendola si arriva al risultato finale

$$\sigma_{\mathrm{L}}^2 = \frac{\frac{\partial w}{\partial T}}{1 - \frac{k}{w}\frac{\partial w}{\partial k} + \frac{1}{4}\big(- \frac{1}{4} - \frac{1}{w} + \frac{k^2}{w^2}\big)\big(\frac{\partial w}{\partial k}\big)^2 + \frac{1}{2}\frac{\partial^2 w}{\partial k^2}}. \tag{D.4}$$

Appendice E

Codici

Andreasen-Huge

Codice E.1: A-H main

```
% - - parte 1 - - %
r=0; S0=100;
N1=101; N2=26;

K = xlsread('SPX','Sheet3','A7:A107');
T = xlsread('SPX','Sheet3','C4:AB4');
DT=ones(N2,1);
for i=1:N2
    if i==1
        DT(i)=T(i);
    else
        DT(i)=T(i)-T(i-1);
    end
end

% volatilita' e prezzi direttamente da mercato
VV = xlsread('SPX','Sheet3','D7:AB107');
impVol=0*ones(N1,N2);
impVol(1:end,2:end)=VV; %e' la superficie di vola implicite.
%Aggiungo una colonna di zeri perche' in t=0 la vola e' 0
clear VV;
% R superficie dei prezzi delle opzioni Call
R=zeros(N1,N2);
for i=1:N2
    for j=1:N1
        R(j,i)=blsprice(S0,K(j),r,T(i),impVol(j,i));
```

```
    end
end

% - - parte 2 - - %
OO=zeros(N2,1);
for i=1:N2
   OO(i)=sum(impVol(:,i)>0);
end
%condizione iniziale per la EDP di Dupire
C=zeros(N1,1);
for j=1:N1
    C(j)=max(S0-K(j),0);
end

sigmaTOT=zeros(N1,N2);
prezziTOTALI=zeros(N1,N2*10);
prezziTOT=zeros(N1,N2);
prezziTOT(:,1)=C;

% - - parte 3 - - %
for i=2:N2
    sigmazero=ones(OO(i),1);
    k=1;
    for j=1:N1
        if impVol(j,i)~=0
            sigmazero(k)=impVol(j,i);
            k=k+1;
        end
    end
    PP=R(:,i);
    CC=C;

    % - - parte 3.1 - - %
    [x,fval]=Runnested(CC,PP,sigmazero,DT(i),K,N1,impVol,i);
    sigmaVET=AndreasenHuge(N1,x,impVol,i);

    % - - parte 3.2 - - %
    C = DiffFin(sigmaVET,DT(i),K,C);

    sigmaTOT(:,i)=sigmaVET;
    prezziTOT(:,i)=C;

    % - - parte 3.3 - - %
```

```
        prezziTOTALI(:,i*10)=C;
        %varia in base al livello di approssimazione scelto
        for j=1:9
            dt=(DT(i)/10)*j;
            D = DiffFin(sigmaVET,dt,K,prezziTOT(:,i-1));
            prezziTOTALI(:,(i-1)*10+j)=D;
        end

    end

    % - - parte 4 - - %
    OO=zeros(N2,1);
    for i=1:N2
       OO(i)=sum(impVol(:,i)>0);
    end
    %errore solo nei prezzi riferiti a volatilita' non nulle
    err=0;
    for i=1:N2
        for j=1:N1
            if impVol(j,i)~=0
                err=err + (R(j,i)-prezziTOT(j,i))^2;
            end
        end
    end
    err= sqrt(err/sum(OO));
```

Codice E.2: A-H Runnested

```
function [x,fval] =  Runnested(CC,PP,x0,DT,K,N1,impVol,i)
% riceve in ingresso i prezzi generati all'istante i-1 (CC)
% e quelli di mercato all'istante i (PP).

[x,fval]=fmincon(@(x) Nestedfun(x,CC,PP,DT,K,N1,impVol,i),...
          x0,[],[],[],[],[],[],@confun);
%restituisce il vettore di volatilita' x
end

% funzione annidata che risolve quella Obbiettivo.
function f = Nestedfun(x,CC,PP,DT,K,N1,impVol,i)

    sigmaVET = AndreasenHuge(N1,x,impVol,i);
    DD = DiffFin(sigmaVET,DT,K,CC);

    err=0;
```

```
    for j=1:N1
        if impVol(j,i)~=0
            err=err + (PP(j)-DD(j))^2;
        end
    end
    f=err;
end
```

Codice E.3: A-H Confun

```
function [C,Ceq] = confun(param)
%  Funzione che contiene i vincoli per la funzione fmincon
% Voglio che i valori delle vola stimati siano tra 0% e 100%
    C=[ -param;
        param-1 ];
    Ceq=[];
end
```

Codice E.4: AndreasenHuge

```
function piecewise=AndreasenHuge(N,sigma,impVol,i)
% prende il vettore delle sigma stimato al tempo t=i
% e restituisce il vettore costante a tratti.

    punti=find(impVol(:,i));
    piecewise=zeros(N,1);
    interv=ones(length(punti),1);
    for k=1:length(punti)-1
        interv(k)=floor((punti(k)+punti(k+1))/2);
    end
    interv(end)=punti(end);

    k=1;
    for j=1:N
        %impVol con soli valori reali
        if j<interv(k)
            piecewise(j)=sigma(k);
        else
            if k<length(punti)
                k=k+1;
            end
            piecewise(j)=sigma(k);
        end
    end
end
```

Codice E.5: A-H DiffFin

```
function Prezzo = DiffFin(sigma,T,K,u)
% calcola un passo temporale della differenze finite.
% riceve in ingresso i prezzi di una slice temporale
% e restituisce quelli della successiva.

    tau=linspace(0,T,2)'; dtau=tau(2)-tau(1);
    dK=2.5;

    g=0.5*(dtau/(dK^2))*(sigma(2:end-1).^2.*K(2:end-1).^2);
    Dd=-g; Nnum=length(u)-1;
    Dm=2*g;
    Du=-g;
    D=spdiags([Dd Dm Du],1:-1:-1,Nnum-1,Nnum-1)';
    A=speye(Nnum-1,Nnum-1)+ D;

    AA=speye(Nnum+1,Nnum+1);
    AA(2:end-1,2:end-1)=A;
    clear A
    AA(2,1)=-g(1); AA(Nnum,Nnum+1)=-g(end);

    u=AA\u;

    Prezzo=u;
end
```

Abasto-Hientzsch-Kust

Codice E.6: A-H-K main

```
% - - parte 1 - - %
r=0; S0=100;
N1=101; N2=26;

K = xlsread('SPX','Sheet3','A7:A107');
T = xlsread('SPX','Sheet3','C4:AB4');
DT=ones(N2,1);
for i=1:N2
    if i==1
        DT(i)=T(i);
    else
        DT(i)=T(i)-T(i-1);
    end
```

```
end

%volatilita' e prezzi direttamente da mercato
VV = xlsread('SPX','Sheet3','D7:AB107');
impVol=0*ones(N1,N2);
impVol(1:end,2:end)=VV; %e' la superficie di vola implicite.
%Aggiungo una colonna di zeri perche' in t=0 la vola e' 0
clear VV;
% R superficie dei prezzi delle opzioni call
R=zeros(N1,N2);
for i=1:N2
    for j=1:N1
        R(j,i)=blsprice(S0,K(j),r,T(i),impVol(j,i));
    end
end
for i=1:N2
    for j=1:N1
        if impVol(j,i)==0
            R(j,i)=NaN;
        end
    end
end

% - - parte 2 - - %
OO=zeros(N2,1);
for i=1:N2
    OO(i)=sum(impVol(:,i)>0);
end
%condizione iniziale per la EDP di Dupire
C=zeros(N1,1);
for j=1:N1
    C(j)=max(S0-K(j),0);
end
prezziTOTmonte=zeros(N1,N2); prezziTOT=zeros(N1,N2);
sigmaTOT=zeros(N1,N2);
prezziTOT(:,1)=C; prezziTOTmonte(:,1)=C;

Nsteps=10;
% - - parte 3 - - %
for i=2:N2
    sigmazero=ones(OO(i),1);
    KK=ones(OO(i),1);

```

```
    k=1;
    for j=1:N1
        if impVol(j,i)~=0
            sigmazero(k)=impVol(j,i);
            KK(k)=K(j);
            k=k+1;
        end
    end
    PP=R(:,i);
    CC=C;

    % - - parte 3.1 - - %
    sigmaKzero=sigmazero.*KK;
    [x,fval]=Runnested(CC,PP,sigmaKzero,DT(i),Nsteps,...
                                    1,impVol,i);
    x=x./KK;

    sigmaVET = AndreasenHuge(N1,x,impVol,i);
    sigmaTOT(:,i)=sigmaVET;

    % - - parte 3.2 - - %
    sigmakappa=sigmaVET.*K;
    C = DiffFin(sigmakappa,DT(i),CC,Nsteps);
    prezziTOT(:,i)=C;

end
% - - parte 4 - - %
Nsteps=100; Nsim=10000;
Z=randn(Nsim,Nsteps*N2);
% genera tutte le simulazioni (anche antitetiche)
% dell'evoluzione del sottostante
[S,Sant]=simGBM(S0,r,sigmaTOT,DT,Nsteps,Nsim,Z,N2,K);

for i=2:N2
    ST=S(:,(i)*(Nsteps));
    STant=Sant(:,(i)*(Nsteps));

    for z=1:N1
        P=max(ST-K(z),0).*exp(-r*T(i));
        Pant=max(STant-K(z),0).*exp(-r*T(i));
        media=(P+Pant)/2;
        prezzo=normfit(media);
        C(z)=prezzo;
```

```
    end
    prezziTOTmonte(:,i)=C+ones(N1,1)*eps;
end

% - - parte 5 - - %
OO=zeros(N2,1);
for i=1:N2
    OO(i)=sum(impVol(:,i)>0);
end
err=0;
for i=1:N2
    for j=1:N1
        if impVol(j,i)~=0
            err=err + (R(j,i)-prezziTOTmonte(j,i))^2;
        end
    end
end
err= sqrt(err/sum(OO));
```

Codice E.7: A-H-K runnested

```
function [x,fval]=Runnested(CC,PP,x0,DT,Nsteps,N1,impVol,i)
% riceve in ingresso i prezzi generati all'istante i-1 (CC)
% e quelli di mercato all'istante i (PP).

[x,fval]=fmincon(@(x) Nestedfun(x,CC,PP,DT,Nsteps,N1,...
             impVol,i),x0,[],[],[],[],[],[],@confun);
%restituisce il vettore di volatilita' x
end

% funzione annidata che risolve quella Obbiettivo.
function f = Nestedfun(x,CC,PP,DT,Nsteps,N1,impVol,i)

    sigmaKappa=AndreasenHuge(N1,x,impVol,i);
    DD = DiffFin(sigmaKappa,DT,CC,Nsteps);

    err=0;
    for j=1:N1
        if impVol(j,i)~=0
            err=err + (PP(j)-DD(j))^2;
        end
    end
    f=err;
end
```

Codice E.8: A-H-K Confun

```
function [C,Ceq] = confun(param)
%  Funzione che contiene i vincoli per la funzione fmincon
% Voglio che i valori delle vola stimati siano tra 0% e 100%
    C= -param;
    Ceq=[];
end
```

Codice E.9: A-H-K DiffFin

```
function Prezzo = DiffFin(sigmaKappa,T,u,Nsteps)
% calcola un passo temporale della differenze finite.
% riceve in ingresso i prezzi di una slice temporale
% e restituisce quelli della successiva.

    tau=linspace(0,T,Nsteps+1)'; dtau=tau(2)-tau(1);
    dK=2.5;

     g=0.5*(dtau/(dK^2))*(sigmaKappa(2:end-1).^2);
     Dd=-g; Nnum=length(u)-1;
    Dm=2*g;
    Du=-g;
     D=spdiags([Dd Dm Du],1:-1:-1,Nnum-1,Nnum-1)';
     A=speye(Nnum-1,Nnum-1)+ D;

    AA=speye(Nnum+1,Nnum+1);
    AA(2:end-1,2:end-1)=A;
    AA(2,1)=-g(1); AA(Nnum,Nnum+1)=-g(end);

    for h=1:Nsteps
         u=AA\u;
    end
    Prezzo=u;
end
```

Codice E.10: simGBM

```
function [S,Sant]=simGBM(S0,r,sigmaTOT,DT,Nsteps,Nsim,Z,N2,K)
% simulo geometric brownian motion
% dS(t)=r*S(t)*dt+sigma(t,S(t))*S(t)*dW(t)

    % inizializzo
     S=zeros(Nsim,Nsteps*(N2)+1); S(:,1)=S0;
     Sant=zeros(Nsim,Nsteps*(N2)+1); Sant(:,1)=S0;
```

```
    for j=2:(Nsteps*(N2))+1
     i=floor(j/Nsteps)+1;
    dt=DT(i)/(Nsteps);

    V=AbastoHientzschKust(sigmaTOT(:,i),S(:,j-1),K);
    Vant=AbastoHientzschKust(sigmaTOT(:,i),Sant(:,j-1),K);
    for y=1:Nsim
        S(y,j)=S(y,j-1).*...
        exp((r-(V(y)^2)/2)*dt +V(y)*sqrt(dt)*Z(y,j-1));

        Sant(y,j)=Sant(y,j-1).*...
        exp((r-(Vant(y)^2)/2)*dt -Vant(y)*sqrt(dt)*Z(y,j-1));
    end
    end
end
```

Codice E.11: AbastoHientzschKust

```
function V = AbastoHientzschKust(sigmaVET,St,K)
% restituisce la volatilita' corrispondente
% allo strike raggiunto dalla simulazione

    V=ones(length(St),1)*sigmaVET(1);

    for i=2:length(K)-1
        ind=(((K(i-1)+K(i))/2)>=St & St>((K(i+1)+K(i))/2));
        V(ind)=sigmaVET(i);
    end
end
```

Gatheral-Jacquier

Codice E.12: G-J main

```
% - - parte 1 - - %
r=0;
N1=101; N2=26;
T = xlsread('SPX','Sheet2','A2:Z2');
DT=ones(26,1);
for i=1:N2
    if i==1
        DT(i)=T(i);
    else
        DT(i)=T(i)-T(i-1);
```

```
    end
end

K = xlsread('SPX','Sheet3','A7:A107');
Fwd = xlsread('SPX','Sheet3','C6:AB6');
Fwd=(Fwd./Fwd(1)).*100;
S0=Fwd(1);
K(end)=0.0001; K=(K./100).*S0;
k=zeros(101,26);
for i=1:26
    k(:,i)=log(K./S0);
end

%volatilita' e prezzi direttamente da mercato
VV = xlsread('SPX','Sheet3','D7:AB107');
impVol=zeros(N1,N2);
impVol(1:end,2:end)=VV; %e' la superficie di vola implicite.
%Aggiungo una colonna di zeri perche' in t=0 la vola e' 0
clear VV;

R=zeros(N1,N2);
for i=1:N2
    for j=1:N1
        R(j,i)=blsprice(S0,K(j),r,T(i),impVol(j,i));
    end
end

% - - parte 2 - - %
Theta=(impVol(61,:).^2).*T;

%calcola in un solo colpo tutti i parametri della superficie
parametri=ones(2,N2);
B1=30; B2=30;
fvalue=ones(B1,B2); %fval
eta=linspace(0,2,B1);
rho=linspace(-1,1,B2);
for i=1:B1 %eta - colonne
    for j=1:B2 %rho - righe
        parametri(1,:)=eta(i); parametri(2,:)=rho(j);
        [~,fval]=runnestedTotal(T,Theta,k,impVol,parametri);
        fvalue(j,i)=fval;
    end
end
```

```
[M,I]=min(fvalue(:));
min_eta=eta(ceil(I/B1));
if mod(I,B2)==0
    min_rho=rho(end);
else
    min_rho=rho(mod(I,B2));
end

% - - parte 3 - - %
parametri(1,:)=min_eta; parametri(2,:)=min_rho;

[x,fval2]=runnestedTotal(T,Theta,k,impVol,parametri);

paramSSVI=zeros(3,N2);
paramSSVI(1,:)=x(1,:);
paramSSVI(2,:)=x(2,:);
paramSSVI(3,:)=ones(1,N2)*0.5;

[W,phi] = SSVI(k,Theta,paramSSVI,1);

alpha=ones(5,N2);
alpha(1,1)=(Theta(1))/2*(1-paramSSVI(2,1)^2);
alpha(2,1)=(Theta(1)*phi(2))/2;
alpha(3,1)=paramSSVI(2,1);
alpha(4,1)=-paramSSVI(2,1)/phi(2);
alpha(5,1)=sqrt(1-paramSSVI(2,1)^2)/phi(2);

% - - parte 4 - - %
surface_v=zeros(N1,N2); surface_local=zeros(N1,N2*10);
surface_l=zeros(N1,N2); WW=zeros(N1,1);
for i=2:N2
    paramEx=paramSSVI(:,i-1);
    alphaEx=alpha(:,i-1);

    % - - parte 4.1 - - %
    [x,fval] = Runnested(T(i),Theta(i),Theta(i-1),k(:,i),...
        impVol(:,i),paramSSVI(:,i),WW,paramEx);

    [WW,phi] = SSVI(k(:,i),Theta(i),x,0);

    paramSSVI(1,i)=x(1);
    paramSSVI(2,i)=x(2);
    paramSSVI(3,i)=x(3);
```

```
alpha(1,i)=(Theta(i)/2)*(1-paramSSVI(2,i)^2);
alpha(2,i)=(Theta(i)*phi)/2;
alpha(3,i)=paramSSVI(2,i);
alpha(4,i)=-paramSSVI(2,i)/phi;
alpha(5,i)=sqrt(1-paramSSVI(2,i)^2)/phi;

% - - parte 4.2 - - %
% passare dalla vola implicita alla vola locale
surface_v(:,i*10)=WW;
surface_local(:,i*10)=WW;

t=(T(i)-T(i-1))/10;
for j=1:10
    % j=T=T(i-1)+j*t  Tn=T(i-1)  Tn+1=T(i)

    tauN=(T(i)-(T(i-1)+j*t))/(T(i)-T(i-1));
    tauNpiu1=((T(i-1)+j*t)-T(i-1))/(T(i)-T(i-1));

    if i==2
          sigmaN=surface_v(:,(i)*10);
    else
          sigmaN=surface_v(:,(i-1)*10);
    end
    sigmaNpiu1=WW;

    surface_local(:,(i-1)*10+j)=tauN*sigmaN+...
                                tauNpiu1*sigmaNpiu1;

    fnEx=alphaEx(2)*alphaEx(3) + (alphaEx(2).*(k(:,i-1)-...
alphaEx(4))./(sqrt((k(:,i-1)-alphaEx(4)).^2 + alphaEx(5)));
    gnEx=(alphaEx(2)*(sqrt((k(:,i-1)-alphaEx(4)).^2 + ...
alphaEx(5))-((k(:,i-1)-alphaEx(4)).^2)./sqrt((k(:,i-1)- ...
alphaEx(4)).^2+alphaEx(5))))./((k(:,i-1)-alphaEx(4)).^2 ...
+ alphaEx(5));
    fn=alpha(2,i)*alpha(3,i) + (alpha(2,i).*(k(:,i)- ...
alpha(4,i)))./(sqrt((k(:,i)-alpha(4,i)).^2 + alpha(5,i)));
    gn=(alpha(2,i)*(sqrt((k(:,i)-alpha(4,i)).^2 + ...
alpha(5,i))-((k(:,i)-alpha(4,i)).^2)./sqrt((k(:,i)- ...
alpha(4,i)).^2+alpha(5,i))))./((k(:,i)-alpha(4,i)).^2 ...
+ alpha(5,i));

    diffT=(sigmaNpiu1-sigmaN)/(T(i)-T(i-1));
    diffX=tauN.*fnEx + tauNpiu1.*fn;
```

```
            diff2X=tauN.*gnEx + tauNpiu1.*gn;

            N=diffT;
            D=1-k(:,i).*diffX./surface_local(:,(i-1)*10+j) +...
            0.25*(-0.25-(1./surface_local(:,(i-1)*10+j)) +...
            (k(:,i)./surface_local(:,(i-1)*10+j)).^2).* ...
            ((diffX).^2) + 0.5.*diff2X;

            if (min(N))<0
                N=N-min(N); %aggiusto numeratore e denominatore
            end
            if (min(D))<0
                D=D-min(D);
            end

            surface_local(:,(i-1)*10+j)=sqrt(N./D);
        end

    end
% - - parte 5 - - %
    for i=1:N2
        surface_l(:,i)=surface_local(:,i*10+1);
    end

%condizione iniziale per la EDP di Dupire
    C=zeros(N1,1); prezziTOT=zeros(N1,N2);
    for j=1:N1
        C(j)=max(S0-K(j),0);
    end
    prezziTOT(:,1)=C; prezziTOTmonte(:,1)=C;
    for i=2:N2
        C=GatheralJacquier(prezziTOT(:,i-1),surface_l(:,i),DT(i),K);
        prezziTOT(:,i)=C;
    end
% Nsim=10000; Nsteps=100;
% % Z = quasiMC(Nsim,Nsteps*N2);
% Z=randn(Nsim,Nsteps*N2);
% conf=zeros(N1,2);
% [S,Sant]=sim_GBM(S0,r,surface_l,T,DT,Nsteps,Nsim,Z,N2,K);
% for i=2:N2
%     ST=S(:,(i)*(Nsteps));
%     STant=Sant(:,(i)*(Nsteps));
%
```

```
%     for z=1:N1
%         payoff=max(ST-K(z),0).*exp(-r*T(i));
%         payoffant=max(STant-K(z),0).*exp(-r*T(i));
%         media=(payoff+payoffant)/2;
%         [prezzo,~,IC]=normfit(media);
%         C(z)=prezzo; conf(z,:)=IC;
%     end
%     prezziTOTmonte(:,i)=C+ones(N1,1)*eps;
% end

% - - parte 6 - - %
OO=zeros(N2,1);
for i=1:N2
    OO(i)=sum(impVol(:,i)>0);
end
%mi serve poi per calcolare l'errore nei soli punti >0
err=0;
for i=10:N2
    for j=1:N1
        if impVol(j,i)~=0
            err=err + (R(j,i)-prezziTOT(j,i))^2;
        end
    end
end
err= sqrt(err/sum(OO));
```

Codice E.13: G-J runnestedTotal

```
function [x,fval] =  runnestedTotal(T,Theta,k,impVol,x0)
options = optimset('fmincon');
options = optimset(options, 'algorithm', 'interior-point');
obbiettivo = @(x) Nestedfun(x,k,impVol,T,Theta);

[x,fval] = fmincon(obbiettivo,x0,...
        [],[],[],[],[],[],@confunTotal,options);
% restituisce la matrice contenente i parametri minimizzati
end

% funzione annidata che risolve la funzione Obbiettivo per
% tutta la matrice. verra' usata poi come guess iniziale
function f = Nestedfun(x,k,impVol,T,Theta)

    eta=x(1,:);
    rho=x(2,:);
```

```
    gamma=ones(1,length(eta))*0.5;

    %calcolo la implied total variance
    param=[eta; rho; gamma];
    [W,~] = SSVI(k,Theta,param,1);
    for i=2:26
        W(:,i)=sqrt(W(:,i)./T(i));
    end

    sum=0;
    for i=1:26
        for j=1:101
            if impVol(j,i)~=0
                sum=sum + (impVol(j,i)-W(j,i))^2;
            end
        end
    end
    f = sqrt(sum);
end
```

Codice E.14: G-J confunTotal

```
function [C,Ceq] = confunTotal(param)
%  Funzione che contiene i vincoli
%  x contiene i parametri che poi minimizzero'
%  su tutta la matrice
%  x(1)=eta  x(2)=rho
C =[-param(1,:); param(2,:)-1; -param(2,:)-1;
    param(1,:).*(1+abs(param(2,:))) - 2;
    ];
Ceq = [];

end
```

Codice E.15: G-J runnested

```
function [x,fval]=Runnested(T,Theta,ThetaEx,k,impVol,x0,...
                    Wold,paramEx)
options = optimset('fmincon');
options = optimset(options, 'algorithm', 'interior-point');
obbiet=@(x) Nestedfun(x,k,impVol,T,Theta,ThetaEx,Wold,paramEx);

[x,fval]=fmincon(obbiet,x0,[],[],[],[],[],[],@confun,options);
%restituisce il vettore contenente i parametri minimizzati
end
```

```
% funzione annidata che risolve la funzione Obbiettivo
function f=Nestedfun(x,k,impVol,T,Theta,ThetaEx,Wold,paramEx)

    eta=x(1);
    rho=x(2);
    gamma=0.5;
    %calcolo la implied total variance
    param=[eta rho gamma];
    [W,~] = SSVI(k,Theta,param,0);

    Cross=Crossedness(W,Wold,k,Theta,ThetaEx,param,paramEx);

    W = sqrt(W./T);
    sum=0;
    for j=1:101
        if impVol(j)~=0
            sum = sum +(impVol(j)-W(j))^2;
        end
    end
    f = sqrt(sum)+100*Cross;
end
```

Codice E.16: G-J runnested

```
function [C,Ceq] = confun(param)
%  Funzione che contiene i vincoli
%  x contiene i parametri che poi minimizzero'
%  sul singolo vettore
%  x(1)=eta  x(2)=rho
     C =[-param(1); param(2)-1; -param(2)-1;
        param(1)*(1+abs(param(2))) - 2;
        ];
    Ceq = [];
end
```

Codice E.17: Crossedness

```
function Cross=Crossedness(Wnow,Wold,k,Theta,ThetaEx,...
                                    param,paramEx)
% Restituisce un vettore contenente i Ci della crossedness.
% Serviranno poi nell'ottimizzazione step-by-step.
    Ki=ones(4,1)*100;
    j=1; I=1;
    while j<5 %cerco le intersezioni
```

```
            if Wnow(I)<Wold(I)
                more=Wold(I:end); less=Wnow(I:end);
            else
                more=Wnow(I:end); less=Wold(I:end);
            end
            i=find((more-less)<0,1);
            if size(I,1)==0
                    break
            else
                    Ki(j)=k(I); %k e' un vettore
                    j=j+1;
            end
    end

    J=find(Ki<100); kT=sort(Ki(J));
    if size(J,1)==1 %creo il vettore dei ktilde
        Ktilde=zeros(2,1);
        Ktilde(1)=kT-2.5; Ktilde(2)=kT+2.5;
    elseif size(J,1)==0
        Ktilde=pi;
    else
        Ktilde=zeros(size(J,1)+1,1);
        Ktilde(1)=kT(1)-2.5; Ktilde(end)=kT(end)+2.5;
        for q=2:size(J,1)
                 Ktilde(q)=(kT(q)+kT(q-1))/2;
        end
    end
    C=0;
    if Ktilde~=pi
        for q=1:length(Ktilde)
        C=max(C,SSVI(Ktilde(q),ThetaEx,paramEx,0)-...
                    SSVI(Ktilde(q),Theta,param,0));
        end
    end
    Cross=C;
end
```

Codice E.18: SSVI

```
function [W,phi] = SSVI(k,Theta,param,FLAG)
%  funzione che calcola la total implied variance
%  dati k, theta e i valori stimati di eta e rho.
    if FLAG==0
        eta=param(1);
```

```
        rho=param(2);
        gamma=param(3);
        phi= eta/( Theta^(gamma)*(1+Theta)^(1-gamma) );

        W=(Theta/2)*(1+rho*phi.*k+sqrt((phi.*k + rho).^2...
                                        +(1-rho^2)));
    else
        eta=param(1,:);
        rho=param(2,:);
        gamma=param(3,:);
        phi= eta./(Theta.^(gamma) .* (1+Theta).^(1-gamma));

        W=zeros(101,26);
        for i=2:26
            W(:,i)=Theta(i)/2.* (1+rho(i)*phi(i).*k(:,i)+...
             sqrt((phi(i).*k(:,i)+rho(i)).^2+(1-rho(i)^2)));
        end
    end

end
```

Bibliografia

[1] Abasto, D., Hientzsch, B., Kust, M., (2013) Monte Carlo Pricing with Local Volatility Grids, *Working paper*.

[2] Andersen, L., Andreasen, J. (1999) Jumping Smiles. *Risk Magazine*, November.

[3] Andreasen, J. (1996) Implied Modelling. *Working paper, Aarhus University, Denmark.*

[4] Andreasen, J., Huge, B. (2011) Volatility interpolation. *Risk Magazine*, March 76-79.

[5] Barucci, E., Marsala, C., Nencini, M., Sgarra, C. (2009) *Ingegneria Finanziaria: Un'introduzione quantitativa,* Egea.

[6] Björk, T. (2004) *Arbitrage theory in continuous time (Second ed.).* Oxford University Press.

[7] Black, F., Scholes, M. (1973) The Valuation of Options and Corporate Liabilities, *Journal of Political Economy*, 637-654.

[8] Boyle, P. (1977) Options: A Monte Carlo Approach, *Journal of Financial Economics,* Vol. 4, 323-338.

[9] Cardano, G., Ferrari, L. (1545) Ars magna o Le Regole dell'Algebra, *Soluzione dell'equazione quadratica.*

[10] Coleman, T., Li, Y., Verma, A. (1998) Reconstructing the unknown local volatility function, *Working paper.*

[11] Cont, R., Tankov, P. (2004) *Financial Modelling with jump processes.* Chapman & Hall/CRC.

[12] Cozzi, D. (2011) Local Stochastic Volatility Models. *Tesi di laurea in Ingegneria Matematica, Politecnico di Milano.*

[13] Crank, J. (1975) *The Mathematics of Diffusion.* Clarendon Press, Oxford.

[14] Derman, E., Kani, I. (1998) Stochastic implied trees: Arbitrage pricing with stochastic term and strike structure of volatility. *International Journal of Theoretical and Applied Finance*, 1(1):61–110.

[15] Dumas, B., Fleming, J., Whaley, R.E. (1998) *Implied Volatility Functions: Empirical Tests.* Journal of Finance, 6.

[16] Dupire, B. (1994) Pricing with a Smile. *Risk*, 7 (1): 18-20.

[17] Dupire, B. (1993) Pricing and Hedging with Smiles, *Proceedings of AFFI Conference, La Baule (also presented at IAFE meeting, New York).*

[18] Fengler, M. R. (2005) Semiparametric modelling of implied volatility. *Lecture Notes, Springer.*

[19] Gatheral, J. (2006) *The Volatility Surface: A Practitioner's Guide.* Wiley Finance.

[20] Gatheral, J. (2004) A parsimonious arbitrage-free implied volatility parameterization with application to the valuation of volatility derivatives. *Presentation at Global Derivatives & Risk Management, Madrid.*

[21] Gatheral, J., Jacquier, A. (2014) Arbitrage-free SVI volatility surfaces. *Quantitative Finance*, 14 (1):59-71.

[22] Gatheral, J., Jacquier, A. (2011) Convergence og Heston to SVI. *Quantitative Finance*, 11 (8):1129-1132.

[23] Glasserman, P. (2004) *Monte Carlo methods in Financial Engineering*, Springer-Verlag.

[24] Guo, G., Jacquier, A., Martini, C., Neufcourt, L. (2013) Generalized Arbitrage-free SVI volatility surfaces. *Working paper at arXiv:1210.7111.*

[25] Heston, S. L. (1993) A Closed-Form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options, *The Review of Financial Studies.*

[26] Hull, J.C. (2006) *Options, futures, and other derivatives.* Pearson Education India.

[27] Jackwerth, J., Rubinstein, M. (1996) Recovering the probability distribution from option prices, *Working paper.*

[28] Jacod, J., Protter, P. E. (2003). *Probability essentials.* Springer Science & Business Media.

[29] JPMorgan (1999) Pricing Exotics under the Smile. *Risk*, November

[30] Le Floc'h, F. (2014) Initial guess for SVI calibration, *Working paper, Calypso Technology.*

[31] Lee, R. (2004) The moment formula for implied volatility at extreme strikes. *Mathematical Finance*, 14(3):469-480.

[32] Lipton, A., Gal, A., Lasis, A., (2013) Pricing of vanilla and first generation exotic options in the local stochastic volatility framework: survey and new results. *Working paper.*

[33] Lipton, A., Sepp, A. (2013) Filling the gaps, *Risk Magazine*, October.

[34] Merton, R.C. (1976) Option pricing when the underlying stock returns are discontinuous. *J. Financ. Econ.*, 3:125-144.

[35] Rebonato, R. (2005) *Volatility and Correlation: The Perfect Hedger and the Fox.* John Wiley & Sons.

[36] Reisinger, C., Calibration of volatility surfaces. *OCIAM, Mathematical Institute Oxford University.*

[37] Rubinstein, M. (1994) Implied binomial tree, *Working paper.*

[38] Seydel, R. (2008) *Tools of Computational Finance.* Springer-Verlag, 4th edition.

[39] Tavella, D., Randall, C. (2000) *Pricing Financial Instruments: The finite difference method.* John Wiley & Sons.

[40] Veneziani, A., Vergara, C. Metodi numerici per le equazioni alle derivate parziali: Differenze finite, *appunti del corso.*

[41] Van der Kamp, R. (2009) Local Volatility Modelling, *a dissertation submitted for the degree of Master of Science.*

[42] Zeliade Systems. (2009) Quasi-explicit calibration of Gatheral's SVI model. *Zeliade White Papers.*

[43] Informazione S&P500, URL:

http://www.borsaitaliana.it/notizie/sotto-la-lente/sp500.htm

[44] Storico S&P500, URL:

http://www.spindices.com/indices/equity/sp-500

[45] Vogt, A. (2006) Post on Wilmott.com, URL:

http:// faculty.baruch.cuny.edu/jGatheral/bloombergsvi2012.pdf