Kalibracja powierzchni local volatility

Marcin Galas, Kamil Krasuski

5 czerwca 2009

Spis treści

1 Opis problemu

1.1 Model Blacka-Scholesa i powierzchnia implied volatility

W standardowym modelu Blacka-Scholesa, mając dane aktywo bazowe o dynamice:

$$\begin{cases} \frac{dS_t}{S_t} = (r-q)dt + \sigma dW_t \\ S_{|t=0} = S_0 \end{cases} \tag{1}$$

możemy wyprowadzić zamkniętą formułę na cenę opcji plain vanilla call na to aktywo:

$$C^{BS}(S,t,K,T,\sigma). \tag{2}$$

Możemy również przeprowadzić operację w pewnym sensie odwrotną, tzn. mając dane kwotowania rynkowe:

$$C^{market}_{K_k,T_l} \ ; k = 1,\dots,k_{max}; \ l = 1,\dots,l_{max},$$

możemy jednoznacznie wyznaczyć zmienności implikowane:

$$\sigma^{I}_{K_k,T_l} \ ; k = 1,\dots,k_{max}; \ l = 1,\dots,l_{max}$$

kalibrujące wyceny z modelu Blacka-Scholesa do wycen rynkowych.

Otrzymane w ten sposób zmienności implikowane okazują się jednak nie być takie same dla wszystkich K_k i T_l (Zad. 9 - estymacja powierzchni implied volatility), co stoi w sprzeczności z założeniami modelu Blacka-Scholesa.

1.2 Model local volatility i powierzchnia local volatility

Adresując powyższy problem rozważa się tzw. model local volatility, w którym:

$$\sigma = \sigma(S,t). \tag{3}$$

W modelu tym, mając dane aktywo bazowe o dynamice:

$$\begin{cases} \frac{dS_t}{S_t} = (r(t)-q(t))dt + \sigma(S,t)dW_t \\ S_{|t=0} = S_0 \end{cases} \tag{4}$$

można rozważać ceny opcji plain vanilla call na to aktywo bazowe:

$$C(S,t,K,T,\sigma(S,t)). \tag{5}$$

Celem naszego projektu będzie, mając daną powierzchnię implied volatility (uzyskaną jako wynik Zadania 9):

$$\sigma^{I}(K_k,T_l) \ ; k = 1,\dots,k_{max}; \ l = 1,\dots,l_{max},$$

dokonanie kalibracji powierzchni local volatility:

$$\sigma(K_k,T_l) \ ; k = 1,\dots,k_{max}; \ l = 1,\dots,l_{max},$$

tak aby ceny europejskich opcji walutowych na kurs EUR/PLN:

$$C(S_0,0,K_k,T_l,\sigma),$$

wyznaczone na podstawie obu powierzchni w odpowiadających im modelach, zgadzały się.

2 Opis algorytmu

2.1 Dupire formula

Jeśli kwotowanych na rynku cen $C^{market}_{K_k,T_l}$ (lub odpowiadających im zmienności implikowanych $\sigma^I(K_k,T_l)$) byłoby wystarczająco dużo (tzn. K_k, T_l pokrywałyby dobrze zakres S i t), to wówczas do wyznaczenia powierzchni local volatility możnaby wykorzystać rezultat teoretyczny uzyskany przez Dupire'a:

$$\sigma(K,T) = \sqrt{2(\frac{C_T + rKC_K}{K^2 C_{KK}})} \tag{6}$$

W praktyce jednak zakres kwotowanych cen rynkowych jest zbyt mały, aby na ich podstawie sensowne było numeryczne wyznaczanie wystepujących we wzorze pochodnych cząstkowych.

Technika rozszerzania zbioru danych rynkowych porzez pewnego rodzaju interpolacje również okazuje się nie dawać dobrych rezultatów przy numerycznym wyznaczaniu powierzchni local volatility w oparciu o formułę Dupire'a.

W związku z tym w pracy wykorzystamy algorytm bazujący na alternatywnych wynikach teoretycznych, zaproponowany przez Turinici'ego.

2.2 Oznaczenia

Mamy dane ceny rynkowe $C(S_0, 0, K_k, T_l, \sigma)$ i odpowiadające im zmienności implikowane $\sigma^I(K_k, T_l)$.

Definiujemy odwzorowanie $\mathcal{P}$ prowadzące ze zmienności lokalnej $\sigma(K_k, T_l)$ w ceny:

$$\mathcal{P}(\sigma)(K_k, T_l) = C(S_0, 0, K_k, T_l, \sigma) \tag{7}$$

oraz odwzorowanie $\mathcal{V}$ prowadzące z wariancji lokalnej ($\nu = \sigma^2$) w wariancję implikowaną ($w = (\sigma^I)^2$):

$$\mathcal{V}(\nu)(K_k, T_l) = w. \tag{8}$$

Zakładamy, że nasz problem posiada (poszukiwane przez nas) rozwiązanie σ_0 i oznaczamy $\nu_0 = \sigma_0^2$. Kalibrację możemy sformułować jako tzw. *inverse problem*:

$$\mathcal{P}(\sigma) = \mathcal{P}(\sigma_0) \text{ lub } \sqrt{\mathcal{V}(\nu)} = \sqrt{\mathcal{V}(\nu_0)}$$

W praktyce zagadnienie tego typu możemy rozwiązać poprzez znalezienie ν minimalizującego odpowiedni funkcjonał kosztowy najmniejszych kwadratów, do którego, w celu zapewnienia dobrego postawienia zadania, dodajemy warunki regularyzacji Tychonoffa:

$$J^\epsilon(\nu) = \sum_{k,l} (\mathcal{V}(\nu)(K_k, T_l) - \mathcal{V}(\nu_0)(K_k, T_l))^2 + \epsilon ||\nabla \nu||^2_{L^2} \tag{9}$$

Minimalizację będziemy chcieli przeprowadzić algorytmem typu Quadratic Programming (QP). Sposób uzyskania poszczególnych elementów potrzebnych do minimalizacji powyższego funkcjonału przedstawimy w kolejnych podrozdziałach.

2.3 Parametryzacja powierzchni $\sigma(K_k, T_l)$

Wprowadzamy dyskretyzację po strike'ach:

$$3.6352 \approx K_1 < \ldots < K_{11} \approx 4.6192; dK \approx 0.0984$$

i po terminach do zapadalności:

$$0.2198 \approx T_1 < ... < T_8 \approx 0.7839; dT \approx 0.0806$$

oraz gęstszą dyskretyzację po cenach aktywa bazowego:

$$3.6352 \approx S_1 < \ldots < S_{41} \approx 4.6192; dS \approx 0.0246$$

i po chwilach czasu:

$$0.2198 \approx t_1 < ... < t_{897} \approx 0.7839; dt \approx 0.0006$$

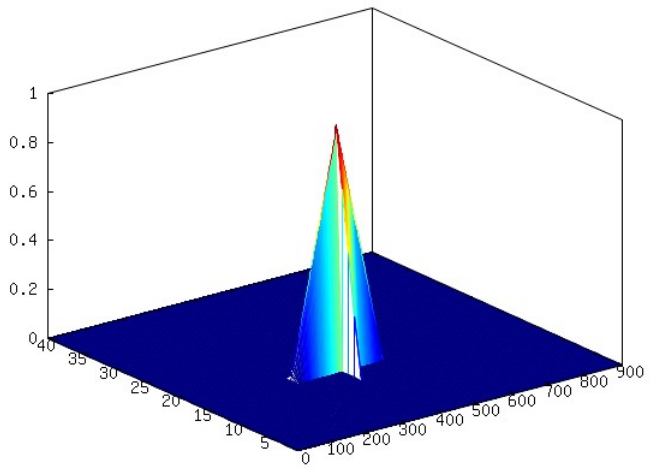
Na tak określonej siatce definiujemy jednoznacznie wyznaczone, kawałkami liniowe, ciągłe funkcje $f_{i,j}(S,t)$ t.że:

$$\begin{cases} f_{i,j}(S,t) = 1 \text{ dla } S = K_i, t = T_j \\ f_{i,j}(S,t) = 0 \text{ dla } S = K_m, t = T_n \text{ gdzie } m \neq i, n \neq j \end{cases} \tag{10}$$

Powierzchnię $\sigma(K_k, T_l)$ będziemy przybliżać kombinacjami liniowymi kształtów $f_{i,j}(S,t)$:

$$\sigma(K_k, T_l) = \sum_{i,j} \alpha_{i,j} f_{i,j}(S,t) \tag{11}$$

Poniżej, dla przykładu, zamieszczamy wykres funkcji $f_{5,4}$:



2.4 Przybliżenie $\mathcal{V}(\nu)(K_k, T_l)$ i macierz $Z^{\nu}_{k,l;i,j}$

Stosujemy przybliżenie pierwszego rzędu wokół punktu ν:

$$\mathcal{V}(\nu + \sum_{i,j} \alpha_{i,j} f_{i,j}(S,t))(K_k, T_l) = \mathcal{V}(\nu)(K_k, T_l) + \sum_{i,j} D_\nu \mathcal{V}(\nu)(K_k, T_l)(f_{i,j})\alpha_{i,j} + o(\alpha) \quad (12)$$

i oznaczamy:

$$Z^{\nu}_{k,l;i,j} = D_\nu \mathcal{V}(\nu)(K_k, T_l)(f_{i,j}) =< D_\nu \mathcal{V}(\nu)(K_k, T_l), f_{i,j} >_{L^2} \quad (13)$$

2.5 Gradient $D_\nu \mathcal{V}(\nu)(K_k, T_l)$

Dla ustalonych K_k, T_l mamy:

$$D_\nu \mathcal{V}(\nu) = D_\nu (\sigma^I)^2 = 2\sigma^I \frac{\partial \sigma^I}{\partial C} D_\nu C \quad (14)$$

Wyrażenie $\frac{\partial \sigma^I}{\partial C}$ jest odwrotnością vegi ze wzoru Blacka-Scholesa:

$$\vartheta^I_{BS} = \frac{\partial C}{\partial \sigma^I} \quad (15)$$

Stąd otrzymujemy:

$$D_\nu \mathcal{V}(\nu)(K_k, T_l) = 2\sigma^I(K_k, T_l) \frac{1}{\vartheta^I_{BS}(K_k, T_l)} D_\nu C(K_k, T_l) \quad (16)$$

2.6 Gradient $D_\nu C(K_k, T_l)$

Cena $C(S, t, K_k, T_l, \sqrt{\nu}) = C^{k,l}$ spełnia dla $S \geqslant 0$ i $t \in [0, T_l]$ równanie Blacka-Scholesa:

$$\begin{cases} \partial_t C^{k,l} + (r-q)S\partial_S C^{k,l} + \frac{\nu S^2}{2}\partial_{SS} C^{k,l} - rC^{k,l} = 0 \\ C^{k,l}(S, T_l) = (S - K_k)^+ \end{cases} \quad (17)$$

Stąd:

$$D_\nu C^{k,l}(S,t) = \frac{S^2}{2}(\partial_{SS} C^{k,l})\chi(S,t) \quad (18)$$

gdzie $\chi(S,t)$ (to samo $\forall k, l$) jest rozwiązaniem równania:

$$\begin{cases} \partial_t \chi + \partial_S((r-q)S\chi) - \partial_{SS}(\frac{\nu S^2}{2}\chi) + r\chi = 0 \\ \chi(S, t=0) = \delta_{S=S_0} \end{cases} \quad (19)$$

Zagadnienie (19) rozwiązujemy odpowiednio dopasowanym schematem Cranka-Nicolson.

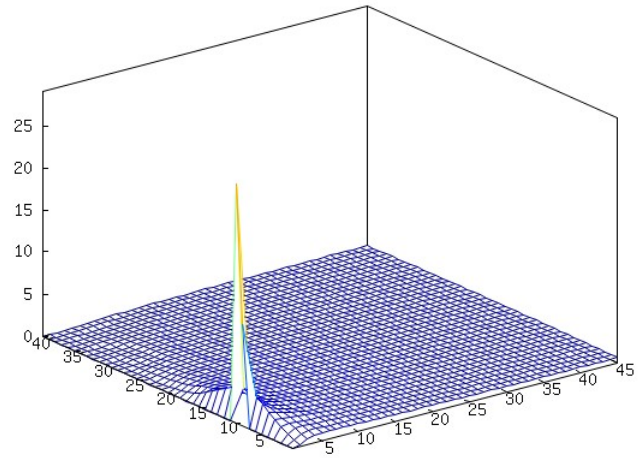
Wyrażenie $\partial_{SS} C^{k,l}$ jest gammą ze wzoru Blacka-Scholesa:

$$\gamma^I_{BS} = \frac{\partial^2 C^{k,l}}{\partial S^2} \quad (20)$$

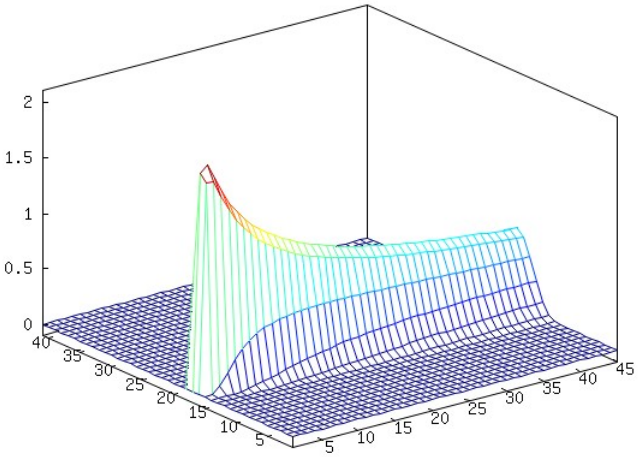
Ostatecznie więc:

$$D_\nu C(K_k, T_l)(S,t) = \frac{S^2}{2}\gamma^I_{BS}(K_k, T_l)\chi(S,t) \tag{21}$$

Poniżej zamieszczamy wykres χ na obciętej siatce:



oraz na pełnej siatce:



2.7 Regularyzacja Tychonoffa - Macierze $(Q_S)_{i,j;k,l}$ i $(Q_t)_{i,j;k,l}$

Ze względu na warunek regularyzacji Tychonoffa $||\nabla\nu||^2_{L^2}$ wyznaczamy macierze:

$$(Q_S)_{i,j;k,l} =< \partial_S f_{i,j}(S,t), \partial_S f_{k,l}(S,t) >_{L^2} \tag{22}$$

$$(Q_t)_{i,j;k,l} =< \partial_t f_{i,j}(S,t), \partial_t f_{k,l}(S,t) >_{L^2} \tag{23}$$

2.8 Ograniczenia przestrzeni rozwiązań

Wprowadzamy ograniczenia dla poszukiwanego rozwiązania:

$$\nu(S,t) \in [\nu_{min}, \nu_{max}] \tag{24}$$

gdzie

$$\nu_{min} = (\frac{1}{2} min\{\sigma^I_{K_k,T_l};\ \text{k=1,...,11; l=1,...,8}\})^2 \tag{25}$$

$$\nu_{max} = (2max\{\sigma^I_{K_k,T_l};\ \text{k=1,...,11; l=1,...,8}\})^2 \tag{26}$$

2.9 Główna pętla algorytmu

1. Dla $\nu^k = \sum_{i,j} \beta_{i,j} f_{i,j}(S,t)$
   ustalamy $B^k = \mathcal{V}(\nu^k)(K,T) - \mathcal{V}(\nu_0)(K,T)$ oraz
   wyliczamy Z^{ν^k}
2. Rozwiązujemy Quadratic Problem (QP):

   $$min_\alpha\{||Z^{\nu^k}\alpha + B^k||^2 + \epsilon < \alpha+\beta, (Q_S+Q_t)(\alpha+\beta) > |\nu_{min} - \beta \leqslant \alpha \leqslant \nu_{max} - \beta\}$$

   Otrzymujemy α^*
3. Przyjmujemy $\nu^{k+1} = \sum_{i,j}(\alpha^*_{i,j} + \beta_{i,j}) f_{i,j}(S,t)$
4. Jeśli $\nu^{k+1} - \nu^k$ jest mniejsze od zadanego błędu przybliżenia, to koniec, w przeciwnym przypadku wracamy do pkt. 1.

3 Dokumentacja programu

3.1 Dane wejściowe

Dane wejściowe składają się z trzech plików. `Zad09_control_data.txt` oraz `Zad09_market_data.txt` zawierają dane wczytywane do zadania 9 (implied volatility). Trzeci plik, `sigma0.txt` jest generowany przez program `implied_volatility.m` i zawiera:
· macierz $sigma0$ - macierz implied volatility o wymiarze 11x8, stanowiącą pierwsze przybliżenie local volatility
· K_local, T_local, dK, dT - opisane poniżej w funkcji `dopasowanie_danych`
· wektory R_d i R_f - wektory odpowiadające stopom procentowym (krajowej i zagranicznej) dla wszystkich tenorów z gęstej siatki

3.2 Opis działania funkcji

`dopasowanie_danych(K_local,T_local,dK,dT,R_d,R_f)`

Parametry:
· K_local - wektor rzadkiej siatki strików o długości 11
· T_local - wektor rzadkiej siatki tenorów o długości 8
· dK - krok po gęstej siatce strików
· dT - krok po gęstej siatce tenorów
· R_d - wektor stóp krajowych (PLN) odpowiadających tenorom z gęstej siatki o długości 1585
· R_f - wektor stóp zagranicznych (EURO) odpowiadających tenorom z gęstej siatki o długości 1585

Zwracane wartości:
· K - wektor gęstej siatki strików o zadanym kroku dK od $K\_local(1)$ do $K\_local(11)$ (o długości 41)
· T - wektor gęstej siatki tenorów o zadanym kroku dT od $T\_local(1)$ do $T\_local(8)$ (o długości 897)
· ind_min - współrzędna rozszerzonego wektora strików gęstej siatki do siatki potrzebnej do Cranka-Nicolson, na której występuje pierwszy strike z gęstej siatki strików
· K_min - minimalny strike w tablicy do liczenia Cranka-Nicolson (rozszerzenie gęstej siatki strików do ≈ 0)
· ind_max - współrzędna rozszerzonego wektora strików gęstej siatki do siatki potrzebnej do Cranka-Nicolson, na której występuje ostatni strike z gęstej siatki strików
· K_max - maksymalny strike w tablicy do liczenia Cranka-Nicolson (rozszerzenie gęstej siatki strików do ≈ 10)
· R_d - wektor stóp krajowych (PLN) odpowiadających tenorom z gęstej siatki

dla wektora T (o długości 897)
· R_f - wektor stóp zagranicznych (EURO) odpowiadających tenorom z gęstej siatki dla wektora T (o długości 897)
· r_d - wektor stóp krajowych (PLN) odpowiadających tenorom z rzadkiej siatki dla wektora T_local (o długości 8)
· r_f - wektor stóp zagranicznych (EURO) odpowiadających tenorom z gęstej siatki dla wektora T_local (o długości 8)
· M - ilość kroków po gęstej siatce srików(41)
· N - ilość kroków po gęstej siatce tenorów (897)
· M_l - ilość kroków po rzadkiej siatce strików (11)
· N_l - ilość kroków po rzadkiej siatce tenorów (8)
· $dK2$ - krok po rzadkiej siatce strików
· $dT2$ - krok po rzadkiej siatce tenorów

Działanie funkcji:
Elementarne operacje nie wymagające wyjaśniania.

`Zagesc_sigma(sgm,K,T,K_l,T_l)`

Parametry:
· sgm - macierz aktualnej zmienności lokalnej o wymiarze 11x8
· K - wektor gęstej siatki strików o długości 41
· T - wektor gęstej siatki tenorów o długości 897
· K_l - wektor rzadkiej siatki strików o długości 11
· T_l - wektor rzadkiej siatki tenorów o długości 8

Zwracane wartości:
Funkcja zwraca macierz o wymiarach 41x897, będącą macierzą zmienności lokalnej odpowiadającej wczytanej macierzy sgm, ale na gęstej siatce.

Działanie funkcji:
Funkcja zagęszcza siatkę pomiędzy węzłami macierzy sgm. Następnie w węzły odpowiadające węzłom macierzy sgm wstawia wartości zmienności zadane macierzą sgm. W kolejnym kroku liniowo interpoluje wartości pomiędzy węzłami w wierszach, dostając 11 wierszy z gęstą siatką, a potem w analogiczny sposób interpoluje wartości w każdej kolumnie. Następnie interpolacja przebiega w odwrotnej kolejności, tzn. najpierw po 8 kolumnach, a potem po wszystkich wierszach. Na koniec dodaje obie powstałe w ten sposób macierze i dzieli przez 2. Dzięki takiej średniej arytmetycznej interpolawane węzły zależą w tej samej mierze od interpolacji po wierszach, jak i kolumnach.

`Crank_Nicolson(So,R_d,R_f,sgm,K_min,K_max,dK,ind_min,ind_max,T,dT)`

Parametry:

· So - cena spot
· R_d - wektor stóp krajowych (PLN) odpowiadających tenorom z gęstej siatki o długości 897
· R_f - wektor stóp zagranicznych (EURO) odpowiadających tenorom z gęstej siatki o długości 897
· sgm - gęsta macierz zmienności lokalnej, otrzymana z funckji `Zagesc_sigma`
· K_min - minimalny strike w tablicy do liczenia Cranka-Nicolson (rozszerzenie gęstej siatki strików do 0)
· K_max - maksymalny strike w tablicy do liczenia Cranka-Nicolson (rozszerzenie gęstej siatki strików do 10)
· dK - krok po gęstej siatce strików
· ind_min - współrzędna rozszerzonego wektora strików gęstej siatki do siatki potrzebnej do Cranka-Nicolson, na której występuje pierwszy strike z gęstej siatki strików
· ind_max - współrzędna rozszerzonego wektora strików gęstej siatki do siatki potrzebnej do Cranka-Nicolson, na której występuje ostatni strike z gęstej siatki strików
· T - wektor gęstej siatki tenorów o długości 897
· dT - krok po gęstej siatce tenorów

Zwracane wartości:
Funkcja zwraca macierz wartości χ, będącą rozwiązaniem równania różniczkowego danego wzorem (19) dla wektora strików i tenorów z rozszerzonej siatki. Macierz ta jest wymiarów 41x897.

Działanie funkcji:
Pierwszym krokiem jest dodatkowe rozszerzenie gęstego wektora strików. Mając zadany krok dK znajduje ona najmniejszy dodatni strike, oraz najmniejszy strike przekraczający 10. Dla tych dodatkowo zdefiniowanych strików przedłuża siatkę sgm, przyjmując, że zmienność na przedłużeniu jest stała. Następnie zadaje warunki brzegowe na tak otrzymanej siatce. Warunek początkowy dla czasu, to delta Diracka dla striku równego So. W przypadku kiedy znajdzie się on pomiędzy węzłami, delta Diracka rozkłada się proporcjonalnie na oba węzły zachowując jednak wartość całki. Warunki brzegowe dla strików wyglądają w sposób następujący: dla $K = K\_max$, $\chi(K,t) = 0$, zaś dla $K = K\_min$, $\chi(K,t) = So - DF(t)K\_min$. Dla tych warunków początkowych funkcja rozwiązuje w sposób standardowy równanie dualne do równania Blacka-Scholesa. Na koniec obcina otrzymaną macierz χ do strików odpowiadających rozszerzonemu wektorowi strików, zwracając tym samym macierz o wymiarach 41x897.

`Gamma_BS(So,r,q,sigma,K,T)` oraz `Vega_BS(So,r,q,sigma,K,T)`

Parametry:
So - cena spot
· r - stopa procentowa krajowych (PLN) odpowiadająca danemu tenorowi opcji

· q - stopa procentowa zagraniczna (EURO) odpowiadająca danemu tenorowi opcji
· $sigma$ - wartość zmienności odpowiadającej danemu tenorowi i strikowi opcji
· K - strike opcji
· T - tenor opcji

Zwracane wartości:
Funkcje zwracają wartości odpowiednio *gammy* i *vegi* z modelu Blacka-Scholesa.

Działanie funkcji:
Wartości obliczane są z podstawowych analitycznych wzorów na współczynniki wrażliwości w modelu Blacka-Scholesa.

```
Wylicz_DvV(K_local,T_local,So,r_d,r_f,sigma,K,T,X)
```

Parametry:
· K_local - wektor rzadkiej siatki strików o długości 11
· T_local - wektor rzadkiej siatki tenorów o długości 8
· So - cena spot
· r_d - wektor stóp krajowych (PLN) odpowiadających tenorom z rzadkiej siatki dla wektora T_local (o długości 8)
· r_f - wektor stóp zagranicznych (EURO) odpowiadających tenorom z gęstej siatki dla wektora T_local (o długości 8)
· $sigma$ - macierz aktualnej zmienności lokalnej o wymiarze 11x8
· K - wektor gęstej siatki strików o długości 41
· T - wektor gęstej siatki tenorów o długości 897
· X - wyliczona z funkcji `Crank_Nicolson` macierz χ o wymiarach 41x897

Zwracane wartości:
Funkcja zwraca macierz czterowymiarową odpowiadającą macierzom gradientów $\mathcal{V}$ dla różnych strików i tenorów z macierzy zmienności lokalnej. Macierz ta ma wymiar 11x8x41x897

Działanie funkcji:
Dla każdego striku i tenoru wyliczane są wartości gammy i vegi. Następnie, korzystając ze wzorów (21) i (16) wyliczamy kolejno macierze DvC i DvV, w którcyh pierwsze dwa indeksy odpowiadają rzadkiej siatce, zaś dwa kolejne gęstej.

```
wylicz_f(i,j,M,N,M_l,N_l)
```

Parametry:
· i, j - indeksy z definicji funkcji $f_{i,j}$
· M - długość rozszerzonego wektora strików (41)
· N - długość rozszerzonego wektora tenorów (897)
· M_l - długość rzadkiego wektora strików (11)

· N_l - długość rzadkiego wektora tenorów (8)

Zwracane wartości:
Funkcja zwracaja macierz odpowiadającą wartościom funkcji $f_{i,j}$ na rozszerzonych siatkach strików i tenorów.

Działanie funkcji:
Funkcja w węzeł macierzy na gęstych siatkach wstawia odpowiadającą wartość z macierzy na rzadkich siatkach (czyli 1). Następnie liniowo opada do 0 do sąsiednich węzłów na rzadkiej siatce (ale tylko wzdłuż osi).

`wylicz_fS(i,j,M,N,M_l,N_l)` oraz `wylicz_ft(i,j,M,N,M_l,N_l)`

Parametry:
· i, j - indeksy z definicji funkcji $f_{i,j}$
· M - długość rozszerzonego wektora strików (41)
· N - długość rozszerzonego wektora tenorów (897)
· M_l - długość rzadkiego wektora strików (11)
· N_l - długość rzadkiego wektora tenorów (8)

Zwracane wartości:
Funkcje zwracaja macierze odpowiadającą wartościom funkcji $\partial_S f_{i,j}$ oraz $\partial_t f_{i,j}$ na rozszerzonych siatkach strików i tenorów.

Działanie funkcji:
Funkcje działają analogicznie, jak funckja `wylicz_f` z tą różnicą, że zamiast liniowej interpolacji wartościami funkcji są wartości pochodnych funkcji $f_{i,j}$. Pochodne te, z racji liniowej interpolacji wartości funkcji $f_{i,j}$, mogą przyjmować wartości jedynie 1 i -1. Dodatkowym założeniem jest przyjęcie, iż $\partial_S f_{i,j}(S_i, t_j) = 0$ i $\partial_t f_{i,j}(S_i, t_j) = 0$.

`wylicz_Z(DvV,M,N,M_l,N_l,dK,dT)`

Parametry:
· DvV - wyliczona wcześniej macierz gradientu zmienności implikowanej w kwadracie (wymiaru 41x897)
· M - długość rozszerzonego wektora strików (41)
· N - długość rozszerzonego wektora tenorów (897)
· M_l - długość rzadkiego wektora strików (11)
· N_l - długość rzadkiego wektora tenorów (8)
· dK - krok po gęstej siatce strików
· dT - krok po gęstej siatce tenorów

Zwracane wartości:
Funkcja zwraca czterowymiarową macierz odpowiadającą iloczynom skalarnym funkcji $D_\nu \mathcal{V}$ oraz funkcji $f_{i,j}$. Dokładniej, komórka o numerze (k, l, i, j) odpowiada

iloczynowi skalarnemu w normie L^2 funkcji $D_\nu\mathcal{V}(\nu)(K_k, T_l)$ oraz $f_{i,j}$. Macierz ta jest wymiaru 11x8x11x8.

Działanie funkcji:
Dla ustalonych k, l, i, j obie funkcje są reprezentowane przez macierze o wymiarach 41x897. Iloczyn skalarny jest liczony poprzez pomnożenie obu macierzy element po elemencie, a następnie przez scałkowanie funkcji reprezentowanej przez otrzymaną macierz po zbiorze określonym przez wektory gęstych strików i tenorów (odpowiada to wysumowaniu objętości prostopadłościanów, których wysokość reprezentowana jest przez wartości wyliczonej macierzy). Ponieważ całkując wzdłuż brzegu nie można otoczyć punktów węzłowych pełnym prostokątem, gdyż druga połowa leżałaby poza macierzą, zbiór po którym całkujemy ten fragment jest dwukrotnie mniejszy. Żeby uwzględnić tę prawidłowość, a nie zmieniać wartości dK i dT odpowiadających za "pole podstawy prostopadłościanów", przed wyliczeniem tej sumy wartości funkcji $f_{i,j}$ zostały podzielone przez 2 wzdłuż każdego brzegu macierzy. W szczególności wartości w węzłach na rogach tej macierzy zostały podzielone przez 4.

`wylicz_QS(M,N,M_l,N_l,dK,dT)` oraz `wylicz_Qt(M,N,M_l,N_l,dK,dT)`

Parametry:
· M - długość rozszerzonego wektora strików (41)
· N - długość rozszerzonego wektora tenorów (897)
· M_l - długość rzadkiego wektora strików (11)
· N_l - długość rzadkiego wektora tenorów (8)
· dK - krok po gęstej siatce strików
· dT - krok po gęstej siatce tenorów

Zwracane wartości:
Funkcje zwracają czterowymiarowe macierze odpowiadające iloczynom skalarnym funkcji $f_{k,l}$ oraz funkcji $f_{i,j}$ (analogicznie jak funkcja `wylicz_Z`). Macierz ta jest wymiaru 11x8x11x8.

Działanie funkcji:
Funkcje działają analogicznie do funkcji `wylicz_Z`. Jedyną różnicą jest to, że zamiast funkcji $D_\nu\mathcal{V}(\nu)(K_k, T_l)$ występuje funkcja $f_{k,l}$.

`wylicz_q(Q,dK2,dT2)`

Parametry:
· Q - macierz powstała przez dodanie do siebie macierzy QS i Qt (wymiar 11x8x11x8)
· $dK2$ - krok po rzadkiej siatce strików
· $dT2$ - krok po rzadkiej siatce tenorów

Zwracane wartości:
Funkcja zwraca dwuwymiarową macierz K taką, że zachodzi równość $< \alpha, Q\alpha >_{L^2} = x' * K * x$, gdzie x jest rozwinięciem macierzy α w wektor. Jest to macierz wymiaru 88x88.

Działanie funkcji:
Funkcja generuje tę macierz zgodnie z obliczeniami przeprowadzonymi poza programem.

`pierwszy_krok(eps,sigma0,Z,QS,Qt,K_local,T_local,dK2,dT2,M_l,N_l)`

Parametry:
· *eps* - parametr określający dokładność obliczeń (0,01)
· *sigma*0 - macierz implied volatility, bedąca pierwszym przybliżeniem local volatility (wymiar 11x8)
· Z - macierz wyliczona z funkcji `wylicz_Z` (wymiar (11x8x11x8)
· QS, Qt - macierze wyliczone z funkcji `wylicz_QS` i `wylicz_Qt` (obie o wymiarze (11x8x11x8)
pozostałe parametry - takie jak w opisie funkcji `dopasowanie_danych`

Zwracane wartości:
Funkcja zwraca:
· macierz $v0$ - macierz *sigma*0 w kwadracie (wykorzystywana później jako punkt odniesienia)
· macierz $v1$ - macierz *sigma*0 w kwadracie (traktowana jako poprzednie przybliżenie)
· macierz $v2$ - macierz nowego przybliżenia zmienności lokalnej w kwadracie (traktowana jako aktualne przybliżenie)
· *blad* - suma modułów różnic macierzy $v2$ i $v1$
· *licznik* - ilość przeprowadzonych przybliżeń metodą minimalizacji (1)
· macierz Q - suma macierzy QS i Qt
· macierz q - macierz z funkcji `wylicz_q`
· v_min - minimalna wielkość kwadratu zmienności lokalnej jaką dopuszczamy w procedurze minimalizacji
· v_max - maksymalna wielkość kwadratu zmienności lokalnej jaką dopuszczamy w procedurze minimalizacji

Działanie funkcji:
Funkcja oblicza macierze Q, *beta*, i B, oraz wartości v_min i v_max zgodnie ze wzorami opisanymi w algorytmie optymalizacyjnym. Następnie w oparciu o wyliczenia wykonane poza programem generuje macierz $z1$ taką, że zachodzi równość: $< Z\alpha, Z\alpha >_{L^2} = x' * z1 * x$, gdzie x jest wektorem z rozwinięcia macierzy α. Dodając do niej pomnożoną przez *eps* macierz q wyliczaną z funkcji `wylicz_q` dostaje się macierz H występującą jako parametr w funkcji qp, która przeprowadza minimalizację. Dalej, również w oparciu o zewnętrzne obliczenia, obliczany jest wektor $q1$ taki, że zachodzi równość: $< \alpha, Q\beta >_{L^2} = x' * q$.

Mnożąc dwukrotność tego wektora przez eps dostajemy wektor w, występujący jako parametr w funkcji qp. Następnie dokonuje się minimalizacja qp, gdzie punktem startowym jest rozwinięta w wektor macierz *beta*. Otrzymany wektor wpisywany jest w macierz, która po dodaniu do niej macierzy beta staje się nowym przybliżeniem kwadratu local volatility ($v2$).

`iteruj(it,eps,v0,v1,v2,blad,licznik,Q,q,v_min,v_max,K,T,K_local,`
`T_local,S0,R_d,R_f,r_d,r_f,sgm,K_min,K_max,ind_min,ind_max,dK,dT,dK2,`
`dT2,M,N,M_l,N_l)`

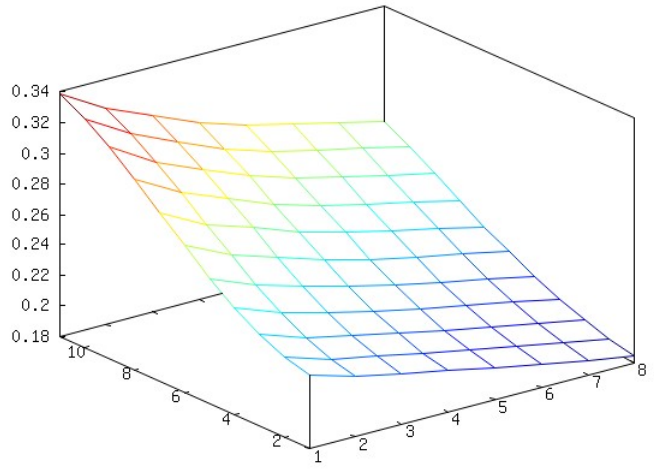
Parametry:
· *it* - maksymalna ilość dopuszczalnych iteracji
pozostałe parametry - takie jak opisane wyżej

Zwracane wartości:
Funkcja zwraca opisane w poprzedniej funkcji $v1$, $v2$, *blad* i *licznik* z ostatniej iteracji.

Działanie funkcji:
Działanie jest analogiczne do funkcji `pierwszy_krok`, różniąc się jedynie uwzględnieniem niezerowej teraz macierzy B i wykorzystując pewne stałe wielkości policzone w funkcji `pierwszy_krok`.

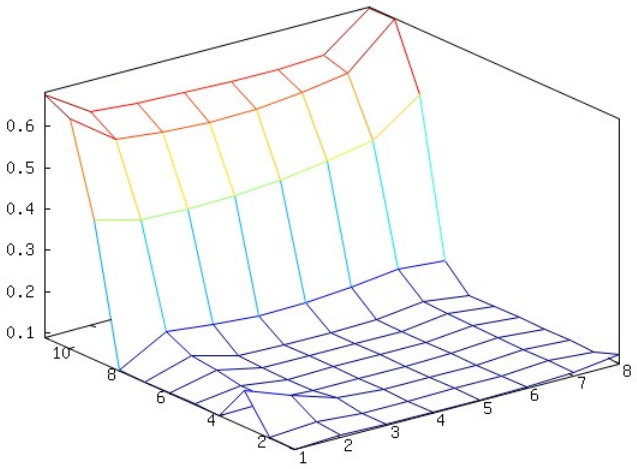
4 Wyniki

Niestety, okazało się, że program nie zwraca oczekiwanych wyników. Dla gładkiej i regularnej płaszczyzny zmienności implikowanej, będącej pierwszym przybliżeniem local volatility (rysunek poniżej), przy próbie drugiego zastosowania funkcji qp program wyświetla błąd.



Wynika to z faktu, iż podczas pierwszej próby minimalizacji optymalne wielkości zmienności znajdowały się poza dopuszczalnym zakresem. W efekcie, jako drugie przybliżenie w znacznej większości użyte zostały wartości z brzegu zbioru dopuszczalnego. Ponieważ zaś algorytm qp startuje z poprzedniego przybliżenia, musiałby startować z brzegu zbioru dopuszczalnego, co powoduje, że program przestaje się liczyć.

Tym samym udało nam się uzyskać jedynie drugie przybliżenie, którego wykres znajduje się poniżej.


0.6
0.5
0.4
0.3
0.2
0.1
10
8
6
4
2
1
2
3
4
5
6
7
8

Literatura

[1] Gabriel Turinici, *Calibration of local volatility using the local and implied instantaneous variance.*

[2] Bruno Dupire, *Pricing with a Smile*, RISK, 7(1):1820, 1994.