Programmi per il calcolo deterministico del prezzo di opzioni call e put

Mariapaola Blancato e Federica Galdelli

Introduzione

Ogetto di questa tesina è l'implementazione del modello di Cox, Ross e Rubinstein per il calcolo del prezzo di opzioni finanziarie *call* e *put*.

Il modello è costruito per un mercato finanziario in cui sono presenti solamente 2 titoli: il titolo non rischioso avente prezzo $S_n^0 = (1+r)^n$ (r è il tasso di interesse), e il titolo rischioso con prezzo S_n. Inoltre si suppone che fra due istanti consecutivi (n e $n+1$) la variazione percentuale del prezzo S_n, che all'istante iniziale ha il valore assegnato S_0, possa assumere solamente due valori a e b con $-1 < a < b$.
Per operare in un mercato completo e privo di arbitraggio si deve imporre, come è stato dimostrato, che valga la seguente relazione: $a < r < b$.

Formula in avanti

In questo primo programma `formuladelta` abbiamo implementato la seguente formula:

$$C_n = (1+r)^{-(N-n)} \sum_{j=0}^{N-n} p^j (1-p)^{N-n-j} F\left(x (1+a)^j (1+b)^{N-n-j}\right) \begin{pmatrix} N-n \\ j \end{pmatrix} \quad (1)$$

la quale ci fornisce il prezzo dell'opzione al tempo n, con $p = \frac{b-r}{b-a}$ (l'unica misura di martingala).
La funzione $F(x)$ è la funzione payoff che, per un prezzo di esercizio k, assume i seguenti valori

- $F(x) = (k-x)_+$, per una opzione put;

- $F(x) = (x-k)_+$, per una opzione call.

Nel programma è stata anche calcolata la funzione *delta*,

$$\Delta(n,x) = \frac{c(n, x(1+b)) - c(n, x(1+a))}{x(b-a)} \tag{2}$$

La *delta* fornisce una misura di copertura per l'opzione considerata, indica, cioè, la sensitività del prezzo rispetto alla variazione del titolo sottostante.

Analizziamo il programma dettagliatamente.

- E' stata costruita una funzione `int` per il calcolo del coefficiente binomiale tramite la formula ricorsiva

$$\begin{pmatrix} n \\ k \end{pmatrix} = \begin{pmatrix} n \\ k-1 \end{pmatrix} \frac{n-k+1}{k} \tag{3}$$

```
int fatt (int n, int k)
{
  int x=1;
  for(int i=1; i<=k; i++)
          x=x*(n-i+1)/i;
  return x;
}
```

- Un'altra funzione serve a calcolare il payoff di un'opzione call:

```
float payoffc (float S, float k)
{
  float p=0;
  if (S>=k) p=S-k;
  return p;
}
```

- Ed infine un'altra funzione `prezzoc` che serve a calcolare il prezzo di un'opzione call:

```
float prezzoc (float x, float a, float b, float p,
               float k, float r, int N, int n)
```

```
{
  float c=0;
  for(int j=0; j<=(N-n); j++)
  c=c+(fatt(N-n,j)*payoffc(x*pow(1+a,j)*pow(1+b,N-n-j),k)
                              *pow(p,j)*pow(1-p,N-n-j));
  c=c*pow(1+r,n-N);
  return c;

}
```

- Inizia,quindi il corpo principale del programma, vengono dichiarate le variabili che saranno utilizzate:

```
main()
{
 float a,b,p,r,S,K,R,sigma,T;
 int N,j,jj,i,m;
```

- Si richiede di inserire l'intero N che rappresenta il numero di sottointervalli in cui dividiamo il tempo.

```
 printf("Inserisci il numero (intero) N\n");
 scanf("%d", &N);
```

- Vengono costruiti tre vetttori di lunghezza N aventi tutti gli elementi uguali a zero e nei quali verranno in seguito inseriti, rispettivamente, i valori dei prezzi (`pr`), i valori della delta (`d`), ed i valori della delta relativi al titolo non rischioso (`dzero`) ad ogni tempo ($0 \leq n \leq N$):

```
 float* pr= new float[N];
 for (i=0; i<=N; i++)
     pr[i]=0;
float* d= new float[N];
 for (i=0; i<=N; i++)
     d[i]=0;
float* dzero= new float[N];
 for (i=0; i<=N; i++)
     dzero[i]=0;
```

- Viene quindi richiesto di inserire, in `input` i valori dei parametri del mercato:

```
printf("Inserisci i parametri. In ordine:\nR, Sigma,T,S,K\n");
 scanf("%f", &R);
 scanf("%f", &sigma);
 scanf("%f", &T);
 scanf("%f", &S);
 scanf("%f", &K);
```

- Si ricavano gli altri parametri: il tasso di intersse $r = \frac{R}{N}$; il valore del salto in basso e quello del salto in alto, rispettivamaente,

$$a = -1 + (1 + r) \exp -\sigma\sqrt{\frac{T}{N}},$$

$$b = (1 + r) \exp \sigma\sqrt{\frac{T}{N}} - 1$$

ed infine la probabiltità del salto in basso $p = \frac{b-r}{b-a}$:

```
r=R/N; /*Tasso d'interesse*/
a=-1+((1+r)*exp((0-sigma)*sqrt(T/N)));
b=((1+r)*exp((sigma)*sqrt(T/N)))-1;
p=(b-r)/(b-a); /*Probabilità del salto in basso: da x a x(1+a)*/
```

- Siama adesso in grado di calcolare i parametri richiesti ad ogni tempo:

```
/*Calcolo prezzo, delta e delta del titolo non rischioso al tempo n,
  li stampo e passo al loro calcolo al tempo n+1*/
 pr[0]=prezzoc(S,a,b,p,K,r,N,0);
 printf("\nPrezzi al tempo 0\n");
 printf("%f", pr[0]);

 for (i=1; i<=N; i++)
 {  for (j=0; j<=i; j++)
        pr[j]=prezzoc(S*pow(1+b,i-j)*pow(1+a,j),a,b,p,K,r,N,i);
    for (jj=0; jj<=i-1; jj++)
    {    d[jj]=(pr[jj]-pr[jj+1])/((b-a)*S*pow(1+b,i-jj-1)*pow(1+a,jj));
```

```
        dzero[jj]= (pr[jj]-pr[jj+1])/(pow(1+r,i));
    }
    printf("\nDelta al tempo %d:\n", i);
    for (j=0; j<=i-1; j++)
        printf("%f   ", d[j]);
    printf("\nDeltazero al tempo %d:\n", i);
    for (j=0; j<=i-1; j++)
        printf("%f   ", dzero[j]);


    printf("\nPrezzi al tempo %d:\n", i);
    for (j=0; j<=i; j++)
    printf("%f   ", pr[j]);

}
```

Formula ricorsiva

Nell'altro programma è stata implementata la seguente formula di ricorrenza per il calcolo del prezzo di un'opzione

$$c(n,x) \quad = \quad \frac{1}{1+r}((1-p)c(n+1,x(1+b)) + pc(n+1,x(1+a))) \quad (4)$$

Vengono utilizzati gli stessi parametri del programma precedente ed in modo analogo sono stati ricavati i valori di a,b, p. Analoghe sono anche le funzioni costruite per il calcolo del coefficiente binomiale, del payoff e del prezzo che verranno utilizzati per calcolo del vettore dei prezzo al tempo finale N

```
for (i=0; i<=N; i++)
    pr[i]=payoffc(S*pow(1+b,N-i)*pow(1+a,i),K);

printf("\nVettore dei prezzi al tempo %d:\n", N);
for (i=0; i<=N; i++)
    printf("%f   ", pr[i]);
```

Si calcola quindi il vettore dei prezzi:

```
/*Calcolo il vettore dei prezzi ad ogni N>n>=0 con la formula ricorsiva.
Lo stampo e lo aggiorno al tempo n-1*/
 for (i=N-1; i>=1; i--)
 {
    for (m=0; m<=i-1; m++)
     {   d[m]=(pr[m]-pr[m+1])/((b-a)*S*pow(1+b,i-m-1)*pow(1+a,m));
         dzero[m]=(pr[m]-pr[m+1])/(pow(1+r,i));
                 /*(pr[m]-d[m]*S*pow(1+b,i-m)*pow(1+a,m))/pow(1+r,i);*/
     }
    printf("\nDelta al tempo %d:\n", i);
    for (j=0; j<=i-1; j++)
        printf("%f   ", d[j]);
    printf("\nDeltazero al tempo %d:\n", i);
    for (j=0; j<=i-1; j++)
        printf("%f   ", dzero[j]);


    printf("\nVettore dei prezzi al tempo %d:\n", i);
     for (j=0; j<=i; j++)
     {   pr[j]=(pr[j]*(1-p)+pr[j+1]*p)/(1+r);
```

```
            printf("%f   ", pr[j]);
        }



     }

    d[0]=(pr[0]-pr[1])/((b-a)*S);
    dzero[0]=(pr[0]-pr[1]);
    printf("\nDelta al tempo %d:\n", 0);
    printf("%f", d[0]);
    printf("\nDeltazero al tempo %d:\n", 0);
    printf("%f", dzero[0]);
    printf("\nVettore dei prezzi al tempo 0:\n");
    pr[0]=(pr[0]*(1-p)+pr[1]*p)/(1+r);
    printf("%f   ", pr[0]);



    printf("\n");
    delete[]pr;
    scanf("%d", &i);
    system("PAUSE");
    return 0;
}
```