QF4102 Financial Modelling and Computation Assignment 2

G01 Wang Zexin, Chen Penghao

October 29, 2017

1 Forward Grid Shooting Method

Write a MatLab function to calculate the American fixed-strike arithmetic-average put option value using Forward Shooting Grid Method with linear interpolation.

1.1 FSGM for American fixed-strike arithmetic-average put option

```
Data: runningTime, $\tau$, $S_0$, $\sigma$, $q$, $A$(running average), $r$, $K$, $N$, $\rho$
Result: $P_{FSGM}$, Option Premium
$\Delta t = \frac{\tau}{N}$; $\Delta x = \sigma\sqrt{\Delta t}$; $\Delta y = \rho\Delta x$;
$u = e^{\Delta x}$; $d = e^{-\Delta x}$; $p = \frac{e^{(r-q)\Delta t} - d}{u - d}$;
for $j = 0$ to $N$ do
    for $k = \frac{-N}{\rho}$ to $\frac{N}{\rho}$ do
        $v(j,k) = max(K - A\exp(k\Delta y))$;
    end
end
for $n = N - 1$ to $1$ do
    for $j = 0$ to $n$ do
        for $k = -\frac{n}{\rho}, -\frac{n}{\rho} + 1, \dots, \frac{n}{\rho}$ do
            $S = S_0 e^{(2j-n)\Delta x}$;
            $A' = Ae^{k\Delta x}$;
            $A_{up} = \frac{A'(n + runningN) + Su}{n + runningN + 1}$;
            $A_{down} = \frac{A'(n + runningN) + Sd}{n + runningN + 1}$;
            $k_u = \left\lfloor \frac{ln(A_{up}/A)}{\Delta y} \right\rfloor$;
            $k_d = \left\lfloor \frac{ln(A_{down}/A)}{\Delta y} \right\rfloor$;
            $v_u = \Pi v(A_{k_u}, S_u)$;
            $v_d = \Pi v(A_{k_d}, S_d)$;
            $v(j,k) = \max(e^{-r\Delta t}(pv_u + (1-p)v_d), K - A')$;
        end
    end
end
$P_{FSGM} = \max(v(0,0), K - A)$;
```

Algorithm 1: Algorithm for pricing American arithmetic-Asian put

The American fixed-strike arithmetic-average put option is initiated 0.25 year ago, and has 0.25 year more to expiry. The underlier price is currently \$100, volatility 40 and dividend yield 0.01 with a running average of \$95, which was taken over the earlier 0.25 year since launched. The risk-free rate r_f is 0.1 and strike price $K = \$100$.

Compute the option value with $\rho = 1$, $\rho = \frac{1}{2}$ and $\rho = \frac{1}{5}$. For each ρ, the number of periods in the lattice is required to be 50, 100, 200 and 400. Compare the computed results and the computation times taken.

We use the forward shooting grid method which on each grid point representing one underlying price and one current time, has an array to represent the option value based on different values of running average. The running average values branch out from the previous running average, which was a weighted average with the starting price of the underlying security. The stock price values branch out from the starting price of the underlying asset.

1.2 Comment on the numerical results and computation times taken for different ρ and N values

The different amoung of time taken and option prices for various set of ρ and N is shown below:

When $\rho = 1$:

ρ	N	Option price	Time taken in μs
1	50	5.3942	0.41758
1	100	5.4301	2.3993
1	200	5.4552	16.218
1	400	5.4685	142.62

When $\rho = 0.5$:

ρ	N	Option price	Time taken in μs
0.5	50	5.36	0.61176
0.5	100	5.4181	4.0884
0.5	200	5.4487	34.326
0.5	400	5.4639	291.35

When $\rho = 0.2$:

ρ	N	Option price	Time taken in μs
0.2	50	5.3533	1.3995
0.2	100	5.412	10.18
0.2	200	5.4446	89.083
0.2	400	5.4487	706.65

Regarding the option value, using the same ρ value, as N increases, the estimated option value increases from $N = 50$ to $N = 400$. The time taken for the program to finish also increase. As N becomes two times the previous value, the run time becomes about 8 times the previous value, implying that the time complexity is cubic with reference to N. This is significant improvement from the two-state-variable BTM, which takes $O(2^n)$ for the computation in each backward iteration.

Across different values of ρ, with the same N value, we observe that the option price obtained is lowered. This is due to the fact that the discretized grids are more finely divided, leading to the payoff at each state being interpolating and landing on a more exact and well-approximated value. Also the run time increases approximately 2 times when we change $\rho = 1$ to $\rho = 0.5$, and approximately 5 times when we change $\rho = 1$ to $\rho = 0.2$, which implies that the time complexity of this method is linear with reference to ρ.

1.3 FSGM for American fixed-strike lookback put option

Next, an American fixed-strike lookback put option is concerned, initiated 0.25 year ago, and still having 0.25 year to expiry. The current underlier price $S_{0.25} = \$1$, volatility $\sigma = 40\%$ and dividend yield $q = 0.01$. The running minimum is at \$0.97 which was taken over the earlier period of 0.25 year. The risk-free rate $r_f = 0.1$ and strike $K = \$0.95$.

```
**Data:** runningTime, $\tau$, $S_0$, $\sigma$, $q$, $A$(running minimum), $r$, $K$, $N$
**Result:** $P_{FSGM}$, Option Premium
$\Delta t = \frac{\tau}{N}$; $\Delta x = \sigma\sqrt{\Delta t}$; $u = e^{\Delta x}$; $d = e^{-\Delta x}$; $p = \frac{e^{(r-q)\Delta t} - d}{u - d}$;
for $j = 0$ to $N$ do
    for $k = \frac{-N}{\rho}$ to $\frac{N}{\rho}$ do
        $v(j,k) = max(K - A\exp(k\Delta x))$;
    end
end
for $n = N - 1$ to $1$ do
    for $j = 0$ to $n$ do
        for $k = -\frac{n}{\rho}, -\frac{n}{\rho} + 1, \ldots, \frac{n}{\rho}$ do
            $S = S_0 e^{(2j-n)\Delta x}$;
            $A' = Ae^{k\Delta x}$;
            $A_{up} = \min(A', Su)$;
            $A_{down} = \min(A', Sd)$;
            $k_u = round(\ln(\frac{A_{up}}{A})/\Delta x)$;
            $k_d = round(\ln(\frac{A_{down}}{A})/\Delta x)$;
            $v_u = v(Ae^{k_u\Delta x}, Su)$;
            $v_d = v(Ae^{k_d\Delta x}, Sd)$;
            $v(j,k) = \max(e^{-r\Delta t}(pv_u + (1-p)v_d), K - A')$;
        end
    end
end
```

Algorithm 2: Algorithm for pricing American lookback put

We use the forward shooting grid method which on each grid point representing one underlying price and one current time. Different from the algorithm for Asian option, we are not using array to represent different values of running minimum on each grid point, and we are not interpolating to obtain to value shot from each grid point. The stock price values branch out from the starting price of the underlying asset.

1.4 Comment on the numerical results and computation times taken for different N values

Running minimum($)	N	Option price	Time taken in μs
0.97	50	0.086453	1.5335
0.97	100	0.081535	10.473
0.97	150	0.08706	35.415
0.97	200	0.090432	89.902
0.97	250	0.092765	169.29
0.97	300	0.087203	287.93
0.97	350	0.089026	444.01
0.97	400	0.090541	646.26
0.97	450	0.091819	923.92
0.97	500	0.092904	1285.7

The option value is at the higher level compared to exercising the option at the current time $t = 0.25$, suggesting that it is still within the holding region. As N increases, we can see that the fluctuations of the option price obtained go smaller and smaller, and becomes stable and converging to the true value.

1.5 Comment on the numerical results obtained from lower running minimum value

Running minimum($)	N	Option price	Time taken in μs
0.57	50	0.38	1.4168
0.57	100	0.38	10.925
0.57	150	0.38	33.6953
0.57	200	0.38	92.7302
0.57	250	0.38	166.4106
0.57	300	0.38	295.1613
0.57	350	0.38	472.1205
0.57	400	0.38	691.3773
0.57	450	0.38	974.2311
0.57	500	0.38	1468.5976

The option values for the running minimum of \$0.57 is always \$0.38, because the current underlying asset price \$1 is too high that it is highly improbable for the future underlier price to be lower than the running minimum, and it is more beneficial to exercise the option immediately at the current time $t = 0.25$. Regardless of the value of N, the option value remains at the same level.

2 Explicit Difference Scheme III for vanilla call option

Statement of the problem

Write a MatLab function to implement the explicit difference scheme III for pricing European vanilla calls. Test the algorithm out with a European vanilla call option with strike $K = \$9$, time to maturity 0.25 year, current underlier price $S_0 = \$9.8$, volatility $\sigma = 0.15$. The dividend yield $q = 1\%$ and risk-free rate is 0.1%.

2.1 Explicit Difference Scheme III for European vanilla call option

```
**Data:** $S_0$, $X$, $r$, $T$, $\sigma$, $N$, $\Delta s$
**Result:** $c_{\text{EDS III}}$, Option Premium
$S_{\max} = 4X$, $\Delta t = \dfrac{T}{N}$, $I = round\left(\dfrac{S_{\max}}{\Delta s}\right)$;
for $j = 0, 1, \ldots, N$ do
    $V_j^0 = (S_{\max} - X)e^{-r(N-j)\Delta t}$;
    $V_j^I = 0$;
end
for $i = 0, 1, \ldots, I$ do
    $V_N^i = \max(X - i\Delta s, 0)$;
    $a_i = \dfrac{1}{1 + r\Delta t}\left[\dfrac{1}{2}\sigma^2 i^2 \Delta t\right]$;
    $b_i = \dfrac{1}{1 + r\Delta t}\left(1 - \sigma^2 i^2 \Delta t - (r - q)i\Delta t\right)$;
    $c_i = \dfrac{1}{1 + r\Delta t}\left[\dfrac{1}{2}(\sigma^2 i^2 + (r - q)i)\Delta t\right]$;
    for $n = N - 1, N - 2, \ldots, 0$ do
        $V_n^i = a_i V_{n+1}^{i-1} + b_i V_{n+1}^{i} + c_i V_{n+1}^{i+1}$ ;
    end
end
$i_0 = round\left(\dfrac{S_0}{\Delta s}\right)$;
$c_{\text{EDS III}} = V_0^{i_0}$;
```

2.2 Comparing option values computed using EDS III and the exact formula

From the explicit scheme III implemented above, with $N = 2900$ and $\Delta s = 0.05$, we obtain the result that the European vanilla call option should be priced at \$$(7.05527 \times 10^{21})$. We computed the option value for the same European vanilla call option using the exact formula, and obtained \$0.82551.

Hence we could see that when $\Delta t = 0.01$, and hence $N = \dfrac{0.25}{0.01} = 25$, the discretized grid will produce ridiculously large option value due to the accumulation of discretization error from each step.

We go further to execute checking of positivity condition, and with $N = 25$, the coefficients b_i are having 653 out of 719 elements violating the positivity condition. This leads to the outcome that the option price is totally off.

2.3 Lower bound value for the coefficients a, b, c to have no violation of positivity conditions

By applying the positivity condition for explicit difference scheme III, i.e by solving the inequality:

$$1 - \sigma^2 i^2 \frac{T}{N} - ri\frac{T}{N} > 0$$

.

Hence a lower bound of N should satisfy:

$$N > \sigma^2 i^2 T + riT$$

$$N > 2916.18$$

.

Since N is an integer, we have the lower bound for N should be 2917.

We verified the result by running the function with $N = 2917$. The option value obatined is \$0.82529, where compared to the theoretical value \$0.82551 obtained using the formula, it has 3 significant figures to be exact. And we further check the violation of positivity conditions, and none of the coefficients a, b, and c got to violate the conditions.

2.4 Cut-off value for the option price to lose all its significant figures

Starting with $N = 2917$, the theoretical value of having no violation of positivity conditions computed in the previous section, we decrease N by 1 each time and re-run the algorithm. The results remain as \$0.82529 as N decreases from 2917 to 2464. As N is lowered to less than 2464, we observe that the results start to deviate from \$0.82529 to some extents. As N decreases further, we observe the following

N	Option value
2455	0.83409
2454	0.81025
2453	0.91772
2452	0.7701
2451	1.1112

Thus we could see that $N = 2453$ is the first time that the results loses all its significant figures. Hence the cut-off value for N should be 2453.

2.5 EDS III for American vanilla call options

By using the lower bound value of N for no violation of positivity conditions, which we obtained was $N = 2917$, we obtained the option price of **\$0.83189**.

Compared to the results obtained earlier of the European styled vanilla call option, which was \$0.82529, the option value for the American vanilla call option is higher. This is due to the extra option to exercise early of American styled options, which adds value to this option.