ap_carr_putamer

Output parameters:

- Price
- Delta

In order to approximate the price of the american putoption, Carr [1] suggests to **randomize the maturity**.
If the maturity is exponentially distributed with parameter $\frac{1}{T}$, the memoryless property of this distribution implies that the passage of time has no effect on the optimal exercise boundary.
Moreover, the price of the randomized american put otion can be compted explicitely by solving an o.d.e.
To reduce the variance, it is better to assume that the maturity is distributed as the sum of N independent exponential variables with parameter $\frac{N}{T}$ (this is an Erlang distribution). Dynamic programmation then provides an explicit formula for the price of this randomized american put option.

/*Pow_int*/
pow_int(x,n) evaluates x^n, wher n is a positive integer.

/*Gamma*/
$$\gamma = \frac{1}{2} - \frac{r-\delta}{\sigma^2}$$

/*DELTA*/
$$\Delta = \frac{T}{n}$$

/*R*/
$$R = \frac{1}{1+r\Delta}$$

/*D*/
$$D = \frac{1}{1+\delta\Delta}$$

/*epsilon*/
$\epsilon = \sqrt{\gamma^2 + \frac{2}{R\sigma^2\Delta}}$

/*p*/
$p = \frac{\epsilon-\gamma}{2\epsilon}$

/*q*/
$q = 1 - p$

/*phat*/
$\hat{p} = \frac{\epsilon-\gamma+1}{2\epsilon}$

/*qhat*/
$\hat{q} = 1 - \hat{p}$

/*Factor*/
This function calculates the factorial of a positive integer.

/*Combi*/
Combi(n,k) evaluates the number $\binom{n}{k}$.

/*Calleuro_n*/
It is the price of the European Call option, with randomized maturity following the Erlang distribution $(n, \frac{n}{T})$.

$$c^{(n)}(s) = \begin{cases} sD^n - KR^n + p^{(n)}(s) & \text{if } s > K \\ \left(\frac{s}{K}\right)^{\gamma+\epsilon} \sum_{k=0}^{n-1} \frac{(2\epsilon \log(\frac{K}{s}))^k}{k!} \sum_{l=0}^{n-k-1} \binom{n-1+l}{n-1} [KD^n \hat{p}^n \hat{q}^{k+l} - KR^n p^n q^{k+l}] & \text{if } s \leq K \end{cases} \tag{1}$$

/*Puteuro_n*/
It is the price of the European Put option, with randomized maturity following the Erlang distribution $(n, \frac{n}{T})$.

$$p^{(n)}(s) = \begin{cases} \left(\frac{s}{K}\right)^{\gamma-\epsilon} \sum_{k=0}^{n-1} \frac{(2\epsilon \log(\frac{s}{K}))^k}{k!} \sum_{l=0}^{n-k-1} \binom{n-1+l}{n-1} [KR^n q^n p^{k+l} - KD^n \hat{q}^n \hat{p}^{k+l}] & \text{if } s \leq K \\ KR^n - sD^n + c^{(n)}(s) & \text{if } s > K \end{cases} \tag{2}$$

/*derivx*/
It computes the partial derivative of a function with respect to its first argument.

/*v*/

for i=1,...,n, $v_i^{(n)} = KR^{n-i+1} - sD^{n-i+1}$

/*A*/

$$
\begin{aligned}
A_i^{(n)}(s;h) \quad = \quad & \sum_{j=h}^{n-i+1} \left(\frac{s}{\underline{s}_{n-j+1}}\right)^{\gamma+\epsilon} \sum_{k=0}^{j-1} \frac{\left(2\epsilon \log\left(\frac{\underline{s}_{n-j+1}}{s}\right)\right)^k}{k!} \sum_{l=0}^{j-k-1} \binom{j-1+l}{j-1} \\
& [p^j q^{k+l} R^j K r - \hat{p}^j \hat{q}^{k+l} D^j \underline{s}_{n-j+1} \delta] \Delta
\end{aligned}
$$

/*b*/

$$
\begin{aligned}
b_i^{(n)}(s) \quad = \quad & \sum_{j=1}^{n-i+1} \left(\frac{s}{\underline{s}_{n-j+1}}\right)^{\gamma-\epsilon} \sum_{k=0}^{j-1} \frac{\left(2\epsilon \log\left(\frac{s}{\underline{s}_{n-j+1}}\right)\right)^k}{k!} \sum_{l=0}^{j-k-1} \binom{j-1+l}{j-1} \\
& [q^j p^{k+l} R^j K r - \hat{q}^j \hat{p}^{k+l} D^j \underline{s}_{n-j+1} \delta] \Delta
\end{aligned}
$$

/*f_1*/
/*f_2*/
/*f_3*/

$$
\begin{aligned}
f_m(\underline{s}_m) = & \sum_{l=0}^{m-1} \binom{m-1+l}{m-1} [KD^m \hat{p}^m \hat{q}^l - KR^m p^m q^l] - A_1^{(m)}(K;2) \\
& - \left(\frac{K}{\underline{s}_m}\right)^{\gamma+\epsilon} [pRKr - \hat{p}D\underline{s}_m \delta] \Delta
\end{aligned}
$$

/*critical_stripped_prices*/

Let $(\tau_i)_{1 \le i \le n}$ be a sequence of independent exponential variables with parameter $\frac{1}{\Delta}$, conditionally on $\tau_1 + ... + \tau_{i-1} \le t < \tau_1 + ... + \tau_i$, the exercise price at time t of the american put option with randomized maturity $\tau_1 + ... + \tau_n$ does not depend on t, and is denoted by $\underline{s}_{n-i+1}$. The constants $\underline{s}_j, 1 \le j \le n$ solve the system $f_j(\underline{s}_j) = 0$, corresponding to the smooth fit condition. We use Newton's algorithm, inductively, to find them.

/*Newton's algorithm*/

It is used to solve the equation $f(x) = 0$, when f is enough regular. We make it recursively:

$$
x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{3}
$$

The first value x_0 must be next to the zero of the function. The recurence stops when $|x_{n+1} - x_n| < \epsilon$, with $\epsilon = 10^{-7}$ here.

/*P1*/
/*P2*/
/*P3*/
The price of the American randomized put option is given by:

$$P^{(n)}(s) = \begin{cases} p_0^{(n)}(s) + b_1^{(n)}(s) & \text{if } s > \underline{s}_0 \equiv K \\ v_i^n(s) + b_i^n(s) + A_i^n(s;1) & \text{if } s \in [\underline{s}_i; \underline{s}_{i-1}], i = 1, ..., n \\ K - S & \text{if } s \leq \underline{s}_n \end{cases} \tag{4}$$

/*decalage*/
This function introduced the strike K as the first argument of an array.

/*Pricing*/
/*Price*/
The price is given by a two points Richardson extrapolation:
$P_A = 2P_2 - P_1$. We use only the two points Richardson extrapolation, because the computation of P3 gives a bad value.
/*Delta*/
To evaluate the delta, we compute: $\frac{P_A(S+h)-P_A(S)}{h}$ with the value 10^{-4} for h.

References

[1] P. CARR. Randomization and the american put. Technical report, Morgan Stanley Bank - New York, 1997. 1