On Pricing of Accumulators

Product nature
The “accumulator” or “accumulative forward” is a daily accumulated and knock-out structured product linked to the performance of an underlying asset. It can be considered as a portfolio of forward contracts with the “occupation time” feature. The accumulated amount of assets depends on the total excursion time of the asset price below the strike price. This leads to an enhanced downside loss. The upside gain is limited by the knock-out feature with an upside barrier.

A typical equality-linked accumulator contract obligates an investor to buy a preset amount of underlying stocks at the strike price $X$, if the closing stock price on a trading day is higher than $X$. However, when the stock closes lower than $X$, the investor has to buy twice the amount of stocks at $X$. Normally, the strike price $X$ is set at a discount of the original spot price $S_0$. This explains why the accumulator is also called “discounted stock” among public. On the other hand, the profit from an accumulator contract is capped by an knock-out barrier $H$ which is set higher than $S_0$.

A sample contract

<table>
<thead>
<tr>
<th>Underlying Shares</th>
<th>SEMBCORP INDUSTRIES LTD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start Date</td>
<td>05 November 2007</td>
</tr>
<tr>
<td>Final Accumulation Date</td>
<td>03 November 2008</td>
</tr>
<tr>
<td>Maturity Date</td>
<td>06 November 2008</td>
</tr>
<tr>
<td>Number of Business Days</td>
<td>250 (subject to adjustment if knocked-out)</td>
</tr>
<tr>
<td>Strike Price</td>
<td>$4.7824</td>
</tr>
<tr>
<td>Initial Price</td>
<td>$5.70</td>
</tr>
<tr>
<td>Knock-out Price</td>
<td>$6.20</td>
</tr>
</tbody>
</table>

Knock-Out Event: A Knock-Out Event occurs if the official closing price of the Underlying Share on any Scheduled Trading Day is greater than or equal to the Knock-Out Price. Under such event, there will be no further daily accumulation of Shares from that day onward. The aggregate number of shares accumulated will be settled on the Early Termination Date, which is the third business day following the occurrence of Early Termination Event.

Shares Accumulation: On each Scheduled Trading Day prior to the occurrence of Early Termination Event, the number of shares accumulated will be 1 when Official Closing Price for the day is higher than or equal to the Strike Price; 2 when Official Closing Price for the day is lower than the Strike Price.

Monthly Settlement Date: The Shares accumulated for each Accumulation Period will be delivered to the investor on the third business day following the end of each monthly Accumulation Period.

Total Number of Shares: Up to the maximum of 500 shares.

Accumulation Period and Delivery Schedule:
<table>
<thead>
<tr>
<th>Accumulation Period</th>
<th>Number of days</th>
<th>Delivery Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>05 Nov 07 to 03 Dec 07</td>
<td>20</td>
<td>06 Dec 07</td>
</tr>
<tr>
<td>04 Dec 07 to 02 Jan 08</td>
<td>19</td>
<td>07 Jan 08</td>
</tr>
<tr>
<td>03 Jan 08 to 04 Feb 08</td>
<td>23</td>
<td>11 Feb 08</td>
</tr>
<tr>
<td>05 Feb 08 to 03 Mar 08</td>
<td>18</td>
<td>06 Mar 08</td>
</tr>
<tr>
<td>04 Mar 08 to 02 Apr 08</td>
<td>21</td>
<td>07 Apr 08</td>
</tr>
<tr>
<td>03 Apr 08 to 02 May 08</td>
<td>21</td>
<td>07 May 08</td>
</tr>
<tr>
<td>05 May 08 to 02 Jun 08</td>
<td>20</td>
<td>05 Jun 08</td>
</tr>
<tr>
<td>03 Jun 08 to 02 Jul 08</td>
<td>22</td>
<td>07 Jul 08</td>
</tr>
<tr>
<td>03 Jul 08 to 04 Aug 08</td>
<td>23</td>
<td>07 Aug 08</td>
</tr>
<tr>
<td>05 Aug 08 to 02 Sep 08</td>
<td>21</td>
<td>05 Sep 08</td>
</tr>
<tr>
<td>03 Sep 08 to 02 Oct 08</td>
<td>21</td>
<td>07 Oct 08</td>
</tr>
<tr>
<td>03 Oct 08 to 03 Nov 08</td>
<td>21</td>
<td>06 Nov 08</td>
</tr>
</tbody>
</table>

12 accumulation periods in total.

**Replication of barrier-type derivatives**

To derive the analytic formula based on discrete settlement of stock transaction on each business day, we assume continuous monitoring of the knock-out barrier $H$.

(i) With immediate transaction of the stock, the payoff at date $t_i$ is given by

\[
\begin{cases} 
0 & \text{if } \max_{0 \leq \tau \leq t_i} S_\tau \geq H \\
S_{t_i} - X & \text{if } \max_{0 \leq \tau \leq t_i} S_\tau < H \text{ and } S_{t_i} \geq X \\
2(S_{t_i} - X) & \text{if } \max_{0 \leq \tau \leq t_i} S_\tau < H \text{ and } S_{t_i} < X 
\end{cases}
\]

Assuming that there are $n$ business days, the fair value of the accumulator is given by

\[
V = \sum_{i=1}^{n} c_{uo}(t_i, X, H) - 2p_{uo}(t_i, X, H),
\]

where $c_{uo}$ and $p_{uo}$ denote the price function of the up-and-out call and up-and-out put, respectively.

Provided that there is no knock-out up to time $t_i$, the payoff at $t_i$ can be decomposed as

\[
\begin{align*}
\text{forward} & = \text{call} + \\times \\
\text{short put} & = \text{put} - \text{call} \\
\end{align*}
\]

The value of the forward is significantly capped by the upper knock-out barrier $H$ while the value of the put sold to the issuer is not much affected by the upper knock-out barrier. The compensation to the buyer is the sale of the stocks at a discount to the prevailing stock price.
Let $T$ denote the settlement date of the stocks fixed at the observation date $t_i$, $T > t_i$, $i = 1, 2, \cdots, n$. Typically, $T$ is 3 business days after the last monitoring date of the given accumulation period. Let $c_{uo}^F(t_i, X, H, T)$ denote the price of a $t_i$-maturity up-and-out barrier call option on a forward contract. The corresponding forward contract is an agreement to buy the underlying stock at time $T$ at a price equals $X$, and a similar definition for $p_{uo}^F(t_i, X, H, T)$ as the price of the corresponding put option. We then have

$$V_{delay} = \sum_{i=1}^{n} c_{uo}^F(t_i, X, H, T) - 2p_{uo}^F(t_i, X, H, T).$$

**Monte Carlo simulation**

We assume the usual Black-Scholes framework where

$$\frac{dS_t}{S_t} = r \, dt + \sigma \, dZ_t.$$

The stock prices at successive time step $t_j$ can be generated as

$$S_j = S_{j-1} e^{\left(r - \frac{\sigma^2}{2}\right) \Delta t + \sigma \epsilon \sqrt{\Delta t}},$$

where $\epsilon \sim N(0,1)$. We take $\Delta t = \frac{T}{N}$, where $N$ is the number of trading days until maturity date $T$.

For each trading day $i$, the following computation is implemented:

1. If $S_i \geq H$, $V_i = V_{i-1} + (1 + K)e^{-ir\delta t}(S_i - e^{-3r\delta t}X)$. Terminate the loop and return $V_N = V_i$; the accumulator is terminated.
2. If $X \leq S_i < H$, $K = K + 1$, $V_i = V_{i-1}$; the number of stock sold is one.
3. If $S_i < X$, $K = K + 2$, $V_i = V_{i-1}$; the number of stock sold is two.
4. If day $i$ is a settlement day, $V_i = V_{i-1} + (1 + K)e^{-ir\delta t}(S_i - e^{-3r\delta t}X)$, $K = 0$; after settlement, the count of stocks accumulated is set to be zero.
5. Go to next day.

Generate $M$ stock price paths do the above computations to obtain $\{V_{N,j}\}_{j=1}^{M}$. The price of the accumulator is:

$$\bar{V} = \frac{1}{M} \sum_{j=1}^{M} V_{N,j}.$$  

The standard error of the estimate is

$$SE_{\bar{V}} = \sqrt{\frac{\text{var}(\{V_{N,j}\}_{j=1}^{M})}{N}},$$

where $\text{var}(\{V_{N,j}\}_{j=1}^{M})$ is the variance of the sample result set. The standard error of the estimate is expected to be inversely proportional to the square root of the number of samples $M$.

**Work elements**

Compute the fair value of the accumulator using

(i) forward shooting grid method;

(ii) Monte Carlo simulation.

Based on the previous experiences by the students of an earlier class, the numerical value of the accumulator is highly dependent on the choices of the time steps and stepwidth in the numerical calculations.
Monte Carlo Simulation Code

```matlab
function [value, se] = acc_mc(S, X, H, r, sigma, settlement, T, M, ngrid)

% Li Lewei, 12 Mar 2010
% MATLAB program for Final Year Project
% Monte Carlo simulation for SEMBCORP accumulator
% S: initial price
% X: strike price
% H: barrier level
% r: risk-free rate
% sigma: volatility
% settlement: array of settlement days
% T: number of trading days per year
% M: number of samples
% ngrid: number of barrier observation per day
% value: the estimated value from Monte Carlo simulation
% se: the standard error of the estimate
% Sample call: [price, stderr] = acc_mc(5.7, 4.7824, 6.1425, 0.02, 0.3, [20, 39, 62, 80, 101, 122, 142, 164, 187, 208, 229, 250], 250, 1000000, 1)

dt = 1/T;
npath = 1000;
nsample = M/npath;
price = zeros(1, nsample);
strikepath = X*ones(1, T);

tic
for i = 1 : nsample
    p = 0;
    for j = 1 : npath
        settle = [0 settlement];
        accumulation = ones(1, T);
        [spath, knockout, discount] = stockpath(S, X, r, sigma, T, ngrid);
        if knockout > 0
            accumulation(knockout:T) = 0;
            Change settlement day as the knockout event occurs.
            for ix = 1:length(settle)-1
                if settle(ix) >= knockout
                    settle(ix) = knockout;
                    settle = settle(1:ix);
                    break
                end
            end
        % Compute the accumulation amount for each day:
        % (0, 1, 2)
        accumulation = (1*(spath>strikepath)+2*(spath<=strikepath))*accumulation;
        % Sum up the PV of payoff for each settlement day
        for k = 1 : length(settle)-1
            p = p + sum(accumulation((settle(k)+1):settle(k+1)))*(spath(settle(k+1))-X)*
            discount(settle(k+1));
        end
    end
    price(i) = p/npath;
end
value = mean(price);
se = sqrt(var(price)/nsample);
toc
return

function [path, knockout, discount] = stockpath(S, X, r, sigma, T, ngrid)
% This function generates random stock paths.
% It returns 'path' as the stock price series, 'knockout' as the
% knockout day, 'discount' as the discount factor series.
N = T*ngrid;
dt = 1/N;
s = zeros(1, N+1);
discount = zeros(1, N);
s(1) = S;
eps = exp((r-0.5*sigma^2)*dt+sigma*sqrt(dt)*randn(1, N));```
knockout = 0;
stayin = 1;
for i = 2 : (N+1)
    s(i) = s(i-1) * eps(1,i-1);
    if ((s(i) > H) && (stayin==1))
        knockout = floor((i-1)/ngrid);
        stayin = 0;
    end
end
path = s(1+ngrid:ngrid:N+1);
discount = exp(-r*dt*(1:N));
return
Forward Shooting Grid

function price = acc_fsgm(S0,X,H,r,sigma,settlement,T,mesh)

% Li Lewei, 12 Apr 2010
% MATLAB program for Final Year Project
% Forward Shooting Grid Method for SEMBCORP accumulator
% S0: initial price
% X: strike price
% H: barrier level
% r: risk-free rate
% sigma: volatility
% settlement: array of settlement days
% T: number of trading days per year
% mesh: number of time steps per day
% Sample call: [price,stderr] = acc_fsgm(5.7, 4.7824, 6.1425, 0.02, 0.3, [20,39,62,80,101,122,142,164,187,208,229,250], 250, 1)
% This is the master function
settlement = [0,settlement];
NT = T;
AccValue = zeros(1,2*NT*mesh+1);
tic
for i = length(settlement):-1:2
    AccValue = fsAccMonthPartmesh(S0,X,H,r,sigma,NT,
    settlement(i),settlement(i)-settlement(i-1),AccValue,
    mesh);
end
price = AccValue;
toc
return

function price = fsAccMonthPartmesh(S0,X,H,r,sigma,NT,Nt,Nday, AccValue,mesh)
% FSGM for one settlement period
dt = 1/(NT*mesh);
discount = exp(-r*dt);
u = exp(sigma*sqrt(dt));
d = 1/u;
p = (exp(r*dt)-d)/(u-d);
S = S0*exp((sigma*sqrt(dt))*(-Nt*mesh:Nt*mesh));
payoff = (S-X)'*(0:2*Nday) + repmat(AccValue',1,2*Nday+1);
V = payoff;
Vtemp = payoff;
indS = Nt*mesh+1;
indj = 1;
for j = Nday-1:-1:0
    % Last time step of each day, daily fixings

for i = (Nday-Nt-j-1)*mesh+1+indS:(Nt-Nday+j+1)*mesh-1+
    indS
    if S(i-1) > X
        Vu = V(i+1,j+1+indj:2*j+1+indj);
        Vd = V(i-1,j+1+indj:2*j+1+indj);
        if S(i+1) >= H
            Vu = (S(i)-X)*(j+indj:2*j+indj);
        end
        if S(i-1) > H
            Vd = (S(i)-X)*(j+indj:2*j+indj);
        end
    else if S(i+1) > X
        Vu = V(i+1,j+1+indj:2*j+1+indj);
        Vd = V(i-1,j+2+indj:2*j+2+indj);
    else
        Vu = V(i+1,j+2+indj:2*j+2+indj);
        Vd = V(i-1,j+2+indj:2*j+2+indj);
    end
end
Vtemp(i,j+indj:2*j+indj) = discount*(p*V(i+1,j+1+indj)+
    (1-p)*V(i-1,j+1+indj));
end
V = Vtemp;
% Other time steps for each day, trivial binomial expectation
if mesh > 1
    for m = 2 : mesh
        for i = (Nday-Nt-j-1)*mesh+m+indS:(Nt-Nday+j+1)*mesh-m+indS
            Vtemp(i,j+indj:2*j+indj) =
                discount*(p*V(i+1,j+indj:2*j+indj)+
                (1-p)*V(i-1,j+indj:2*j+indj));
        end
        V = Vtemp;
    end
end
price = V((Nday-Nt)*mesh+indS:(Nt-Nday)*mesh+indS,indj)';
return
Crank-Nicolson Scheme Code

function price = acc_cn(S0, X, H, r, sigma, settlement, NT, Ns, Mesht)
    % Li Lewei, 15 Apr 2010
    % MATLAB program for Final Year Project
    % Crank-Nicolson scheme for SEMBCORP accumulator
    %
    % S0: initial price
    % X: strike price
    % H: barrier level
    % r: risk-free rate
    % sigma: volatility
    % settlement: array of settlement days
    % NT: total number of days per year
    % Ns: number of steps in S mesh
    % Mesht: number of time steps per day
    %
    % Sample call: price = acc_cn(5.7, 4.7824, 6.1425, 0.02, 0.3, 
    % [20, 39, 62, 80, 101, 122, 142, 164, 187, 208, 229, 250], 250, 1000, 32)
    %
    Smax = 1.25*H;
    %
    % settlement = [20, 39, 62, 80, 101, 122, 142, 164, 187, 208, 229, 250];
    %
    % NT = settlement(length(settlement));
    % AccValue = zeros(Ns-1,1);
    tic
    if length(settlement)>1
        for i = length(settlement):-1:2
            AccValue = acc_cn_month(S0, X, H, r, sigma, NT,
                settlement(i), settlement(i)-settlement(i-1),
                AccValue, Smax, Ns, Mesht);
        end
    end
    AccValue = acc_cn_month(S0, X, H, r, sigma, NT, settlement(1),
        settlement(1), AccValue, Smax, Ns, Mesht);
    price = AccValue(floor(Ns*S0/Smax));
    toc
    return

function price = acc_cn_month(S0, X, H, r, sigma, NT, Nt, Nday, AccValue,
    Smax, Ns, Mesht)
    % CN for one settlement period
    h = Smax / Ns;
    dt = 1 / (NT*Mesht);
    indX = floor(X/h);
    indH = floor(H/h);
    S = [1:Ns-1]*h;
    payoff = (S-X)'*(0:2*Nday) + repmat(AccValue, 1, 2*Nday+1);
    % boundary conditions
47 \ u = \text{payoff};
48 \ u0 = (-X)*\exp(-r*[Nday*Mesht:-1:0]*dt)'*(0:2*Nday)+((-X)*\exp(-r
 *[Nday*Mesht:-1:0]*dt).*floor((Nday*Mesht:-1:0)/Mesht))'*ones
 (1,2*Nday+1);
49 \ u1 = (H-X)*\exp(0*[Nday*Mesht:-1:0]*dt)'*(0:2*Nday);
50
% matrix build-up
51 i = [1:Ns-1]';
52 alpha = sigma^2 / 2 * (i.^2);
53 beta = r / 2 * i;
54 B = spdiags([alpha + beta, -alpha * 2 - r + 2/dt, alpha - beta],
 [-1:1], Ns-1, Ns-1)';
55 A = spdiags([-alpha - beta, alpha * 2 + r + 2/dt, -alpha + beta ],
 [-1:1], Ns-1, Ns-1)';
56
57 uu0 = (alpha(1) - beta(1)) * (u0(1:Nday*Mesht,:)) + u0(2:Nday*
 Mesht+1,:));
58 uu1 = (alpha(Ns-1) + beta(Ns-1)) * (u1(1:Nday*Mesht,:)) + u1(2:
 Nday*Mesht+1,:));
59
60 count = 0;
61 for n = 1:Nday
62 \ u(1:indX,(1+Nday-n):(1+2*(Nday-n))) = u(1:indX,(3+Nday-n :
 )(3+2*(Nday-n)));
63 \ u(indX+1:indH,(1+Nday-n):(1+2*(Nday-n))) = u(indX+1:indH
 ,(2+Nday-n):(2+2*(Nday-n)));
64 if n>1
65 \ u(indH+1:Ns-1,(1+Nday-n):(1+2*(Nday-n))) = (S(
 indH+1:Ns-1)-X)'*((Nday-n):(2*(Nday-n)));
66 end
67 for m = Mesht:-1:1
68 \ w = B * u;
69 \ w(1,:) = w(1,:) + uu0(1+n*Mesht-m,:);
70 \ w(Ns-1,:) = w(Ns-1,:) + uu1(1+n*Mesht-m,:);
71 \ u = A \ w;
72 end
73 end
74 price = u(:,1);
75 return