



(-)ansa



```

/*****
Copyright ©2009, Kuberre Systems, Inc. All rights reserved. For more information, address Kuberre Systems, Inc,
805 Turnpike Street, North Andover, MA 01845.
*****/

```

```

% Computes the Boyle (1986) Trinomial Tree for American Call/Put Option
Values based
% on the following inputs using the computational power of HANSA:
% CallPut          =      Call = 1, Put = 0
% AssetP           =      Underlying Asset Price
% Strike           =      Strike Price of Option
% RiskFree         =      Risk Free rate of interest
% Div              =      Dividend Yield of Underlying
% Time             =      Time to Maturity
% Vol              =      Volatility of the Underlying
% nSteps           =      Number of Time Steps for Trinomial Tree to
take
% Please note that the use of this code is not restricted in anyway.
% However, referencing the author of the code would be appreciated.
% To run this program, simply use the function defined in the 1st line.
% http://www.global-derivatives.com
% info@global-derivatives.com
% Kevin Cheng (Nov 2003)

```

```

function [] = TrinomialAmerican_hansa(CallPut, AssetP, Strike, RiskFree, Div,
Time, Vol, nSteps)

```

```

dt = Time / nSteps;          % Allocates the time steps
cc = RiskFree - Div;         % Specifies the cost of carry (r - D)

```

```

if CallPut
    create_matrix('b', 1, 1, 1, 3, 1, '', 0);
end
if ~CallPut
    create_matrix('b', 1, 1, 1, 3, -1, '', 0);
end

```

```

create_matrix('AssetP', 1, 1, 1, 3, AssetP, '', 0);
create_matrix('Strike', 1, 1, 1, 3, Strike, '', 0);

```

```

exp(RiskFree * dt, 'RR')

```

```

% The magnitude of an up movement
exp(Vol * sqrt(2 * dt), 'Up');
exp(Vol * sqrt(2 * dt), 'Down');

```

```

% The magnitude of a down movement
power('Down', -1)

```

```

%%% Specifies the probability of up, down and mid moves for trinomial tree
exp(cc * dt / 2, 'PUP1');

```



Hansa



```
exp(-Vol * sqrt(dt / 2), 'PUP2');
sub('PUP1', 'PUP2');
exp(Vol * sqrt(dt / 2), 'PUP3');
exp(-Vol * sqrt(dt / 2), 'PUP4');
sub('PUP3', 'PUP4');
power('PUP4', -1);
multiply('PUP2', 'PUP4', 'P_up');
power('P_up', 2);
unload('PUP1');
unload('PUP2');
unload('PUP3');
unload('PUP4');

exp(Vol * sqrt(dt / 2), 'PDN1');
exp(cc * dt / 2, 'PDN2');
sub('PDN1', 'PDN2');
exp(Vol * sqrt(dt / 2), 'PDN3');
exp(-Vol * sqrt(dt / 2), 'PDN4');
sub('PDN3', 'PDN4');
power('PDN4', -1);
multiply('PDN2', 'PDN4', 'P_down');
power('P_down', 2);
unload('PDN1');
unload('PDN2');
unload('PDN3');
unload('PDN4');

ones(1,1,'mid');
copy('P_up', 'pup');
copy('P_down', 'pdn');
sub('mid', 'pup');
sub('pup', 'pdn');
copy('pdn', 'P_mid');
unload('mid');
unload('pdn');
unload('pup');

exp(-RiskFree * dt, 'Df');

% Sets up the asset movements on the trinomial tree
zeros(1,2*nSteps, 'Value');
for i = 0:(2 * nSteps)
    State = i + 1;
    copy('Up', 'U');
    copy('Down', 'D');
    copy('Strike', 'S');
    power('U', max(i - nSteps, 0));
    power('D', max(nSteps * 2 - nSteps - i, 0));
    multiply('AssetP', 'U', 'TMP1');
    multiply('TMP1', 'D', 'TMP2');
    sub('TMP2', 'S');
    multiply('b', 'S', 'Res');
```



Hansa



```
movement = max(0, getElement('Res',1,1));
setElement('Value', 1, 1, State, State, movement);
unload('U');
unload('D');
unload('S');
unload('TMP1');
unload('TMP2');
unload('Res');
end

% Works backwards to determine the price of the option
for TT = (nSteps - 1):-1:0
    for i = 0:(TT * 2)
        State = i + 1;
        copy('Up','U');
        copy('Down','D');
        copy('Strike','S');
        power('U', max(i - TT, 0));
        power('D', max(TT * 2 - TT - i, 0));
        multiply('AssetP', 'U', 'TMP1');
        multiply('TMP1', 'D', 'TMP2');
        sub('TMP2', 'S');
        multiply('b', 'S', 'Res');
        price = max((getElement('P_up',1,1) * getElement('Value',1,(State + 2)) +
            getElement('P_mid',1,1) * getElement('Value',1,(State + 1)) +
            getElement('P_down',1,1) * getElement('Value',1,State)) *
            getElement('Df',1,1), getElement('Res',1,1));
        setElement('Value',1,1,State,State,price);
        unload('U');
        unload('D');
        unload('S');
        unload('TMP1');
        unload('TMP2');
        unload('Res');
    end
end

Trinomial = getElement('Value',1,1)
```