BackToBasics.UsingNDSolve.nb

American - style options in the Black - Scholes model
Handle discrete dividends, using the' Back to Basics' piecewise GBM theory and NDSolve.
Handles any Div policy function

Example for Vol II, Ch .9
Created under Ver 9.0
Created Mar 7, 2016 by alan

Documentation.
For code explanantions, further documentation, see
"Option Valuation under Stochastic Volatility II: with Mathematica Code",
Alan L. Lewis, (2016) Finance Press, Newport Beach, California
Chapter 9 : "Back to Basics: An Update on the Discrete Dividend Problem"

Notes :
Vplus (S) = P (S, td -)
Vminus (S) = P (S, td +)

Important : Early exercise at intra - dividend times, is checked every dt = 1/nexoppsperyear,
where the latter is the `number of exercise opportunities per year'. (Using 250 mostly here).
When choosing ex - dividend times, you MUST choose times that are integer multiples of dt, or they will be missed by the solver

```
Clear[MIU, MMU];
MIU := N[MemoryInUse[] / 10^9];
MMU := N[MaxMemoryUsed[] / 10^9];


(* Every dt=1/nexoppsperyear,
check for soln falling below initi soln, and adjust *)
(* The PDE time τ = T - t, where t = calendar time *)
(* The first dividend is at td1 in pde time *)
(* The interval between dividends is tdint in pde time *)
(* So there is a discrete dividend Div,
whenever the calendar time t=T-td1, T-td1-tdint, T-td1-2 tdint, etc *)
(* So, there may be 0 or multiple dividends per year *)
(* The solver is forced to take at least 10 tsteps between events *)
(* Uses global DivPolicy[Div] *)
Clear[solverDivPolicy];
solverDivPolicy[S0_, K_, td1_, tdint_, Div_,
  nexoppsperyear_, r_, initsoln_, T_, npts_, rpts_, pflag_] :=
 Module[{Smap, xmap, xgrid, Soln, h, xmin, xmax, dt, ecnt = 0,
```

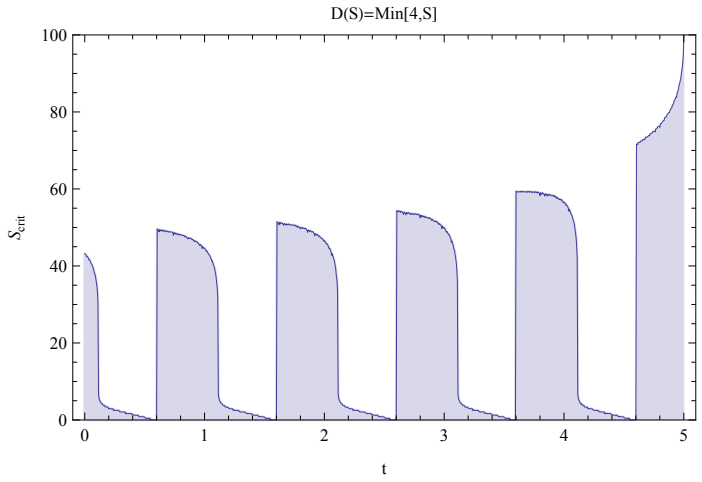
```
 divflag, divecnt = 0, nextTd = td1, hsoln, DP},

 xgrid = N[Table[i / npts, {i, 0, npts - 1}]]; (* Keep away from 1 *)
 xmin = xgrid[[1]];
 xmax = xgrid[[npts]];
 dt = N[1 / nexoppsperyear];

 Smap[x_] := S0 x / (1.0 - x);
 xmap[s_] := s / (s + S0);

 DP[S_] = DivPolicy[Div, S];
 Print["Solver: using div policy D(S)=",
  DP[S], " td1=", td1, " tdint=", tdint, " T=", T];

 Clear[Soln, h];
 Soln =
  h /. NDSolve[{∂τh[x, τ] == a1[x] ∂x,xh[x, τ] + b1[x] ∂xh[x, τ] - r h[x, τ],
      h[x, 0] == initsoln[x],
      h[xmin, τ] == initsoln[xmin], h[xmax, τ] == initsoln[xmax],
       WhenEvent[Mod[τ, dt] == 0, ecnt++; divflag = 0;
       If[(divecnt == 0 && Chop[τ - td1] == 0) || (divecnt > 0 && Chop[nextTd - τ] == 0),
        Print["Div detected at τ=", τ];
        divflag = 1; divecnt++; nextTd += tdint, Null];
       hsoln[x_] := If[divflag == 1, h[xmap[Smap[x] - DivPolicy[Div, Smap[x]]], τ],
         Max[h[x, τ], initsoln[x]]];
       h[x, τ] → Outer[hsoln[#1] &, xgrid]]}, {h}, {x, xmin, xmax},
     {τ, 0, T}, (* MUST USE {τ,0,T} *)
     StepMonitor :→ (ct++; If[pflag == 1 && Mod[ct, rpts] == 0,
        Print["solverEvent: ct=", ct, " τ=", τ, " MIU(GB)=", MIU], Null]),
     MaxSteps → 1 000 000, MaxStepFraction → Min[dt / (10 T), 1 / npts],
      Method → {"MethodOfLines",
       "SpatialDiscretization" → {"TensorProductGrid", "Coordinates" → xgrid}}][[
    1]];

 Print["solverDivPolicy: done; T=", T, " found ", ecnt,
  " early exercise events including ", divecnt, " discrete dividend events"];
 Return[Soln]]


(* American-style Put Values under BS Model *)
(* Uses NDSolve on unit square coords with WhenEvent *)
(* Allows multiple discrete dividends Div, specifying (Td1,Tdint), where *)
(* The first dividend prior to expiration is at T-Td1,
and the interval between dividends is Tdint *)
(* noppsperyear = number of Bermudan exercise opps per year; suggest 250 *)
(* You MUST have ex-div dates lying exactly on a Bermudan exercise time *)
(* So, if noppsperyear = 1/250 = 0.004,
then ex-div times must be exactly divisible by 0.004 *)
(* Requires Td1 < T; but if you want no dividends, set Div=0 *)
```

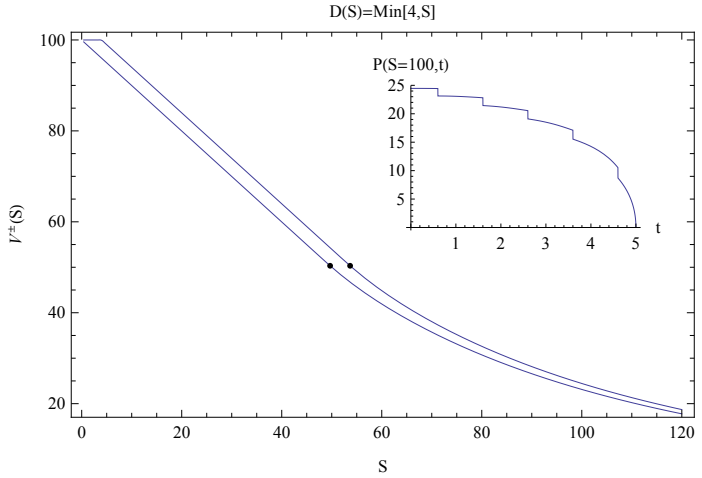
```
Clear[PutDivPolicy];
PutDivPolicy[S0_, K_, T_, r_, sig0_, pflag_, rpts_, solverpflag_,
  npts_, noppsperyear_, Td1_, Tdint_, Div_, tinset_, size_] :=
  Module[{C0, xmin, xmax, Smap, V = sig0^2, initsoln, soln, ans, eps, divx, dt,
            xxgrid, xmap, x0, x1, t1, ans0, MyTimeValue, S1, teval, Scrit,
            p, plotinset, point1, point2, label, Tdlast},

   dt = 1 / (10 noppsperyear); (* Max dt that the solver will use *)
   If[T == 0, Return[Max[K - S0, 0]], Null];

   (* Unit square coords *)
   C0 = N[S0];
   xmap[s_] := s / (s + C0);

   Clear[Smap];
   Smap[x_] := C0 x / (1.0 - x);

   eps = 1.0 / npts;
   xmin = eps;
   xmax = 1.0 - eps;

   (* PDE coefs for x-derivatives: GLOBALS *)
   Clear[a1, b1];
    a1[x_] = Simplify[0.5 V x^2 (1.0 - x)^2];
    b1[x_] = Simplify[(r - V x) x (1.0 - x)];

   (* Hotspot *)
    x0 = 0.5;

   (* Insert the dividend policy function here, with chart label *)
   Clear[DivPolicy];  (* Global *)

    DivPolicy[divx_, Sx_] = Min[divx, Sx];  (* HHL Liquidator *)
    label = "D(S)=" <> "Min[" <> ToString[Div] <> ",S]";

   (* DivPolicy[divx_,Sx_]= If[Sx<divx,0,divx];   (* HHL Survivor *)
    label= "D(S)="<>  ToString[Div] <> " × 1(S>"<>ToString[Div]<>")"; *)

   (* initial soln *)
   initsoln[x1_] = Max[K - Smap[x1], 0];

   Off[NDSolve::eerri];
   Clear[ct]; ct = 0;
    soln = solverDivPolicy[S0, K, Td1, Tdint,
     Div, noppsperyear, r, initsoln, T, npts, rpts, solverpflag];

   ans0 = soln[x0, T];
   Print["PutDivPolicy:ct=", ct, " x0=", x0, " npts=", npts,
    " noppsperyear=", noppsperyear, " Price=", ans0, " (MMU=", MMU, " GB)"];
```

```
If[pflag == 0, Return[ans0], Null];

(* Plot soln starting from t=0 to tinset *)
plotinset =
 Plot[soln[x0, T - t1], {t1, 0, Min[tinset, T]}, AxesLabel → {"t", "P(S=100,t)"},
  ImageSize → 150];

MyTimeValue[x1_, t1_] := soln[x1, t1] - Max[K - Smap[x1], 0];
xxgrid = N[Table[i / npts, {i, 0, npts - 1}]]; (* Keep away from 1 *)

Clear[GenericTV, GenericScrit, critdata];
GenericTV[x1_, t1_] := MyTimeValue[x1, T - t1];
GenericScrit[t1_] := If[t1 == T, K, Smap[GenericXcritSearch[xxgrid, t1]]];
Print["At t=0 found Scrit=", GenericScrit[0]];

(* Now at tD-, inst prior to going ex-dividend for the last time,
counting backwards from expiration *)
Tdlast = Td1;
While[Tdlast ≤ T, Tdlast += Tdint];
Tdlast -= Tdint;

(* Now at tD-, inst prior to going ex-dividend for the last time,
counting backwards from expiration *)
teval = Tdlast + dt;
Scrit = GenericScrit[T - teval];
Print["At t=", T - teval, " with dx=", eps, " found Scrit=", Scrit];

Print["Creating Vpluschart at t=", T - teval];
Clear[Vpluschart];
Vpluschart =
 Plot[soln[xmap[S1], teval], {S1, Smap[xmin], 1.2 K}, ImageSize → size,
  FrameLabel → {{"V±(S)", Null}, {"S", label}}, Frame → True,
   Epilog -> Inset[plotinset, {87, 75}]];

(* Now at tD+, just after going ex-dividend *)
teval = Tdlast - dt;
Print["Creating Vminuschart at t=", T - teval];
Clear[Vminuschart];
Vminuschart =
 Plot[soln[xmap[S1], teval], {S1, Smap[xmin], 1.2 K}, ImageSize → size,
  FrameLabel → {{"V±(S)", Null}, {"S", label}}, Frame → True];

Scrit = GenericScrit[T - teval];
Print["At t=", T - teval, " with dx=", eps, " Scrit=", Scrit];

point1 = Point[{Scrit, K - Scrit}];
point2 = Point[{Scrit + Div, K - Scrit}];
```

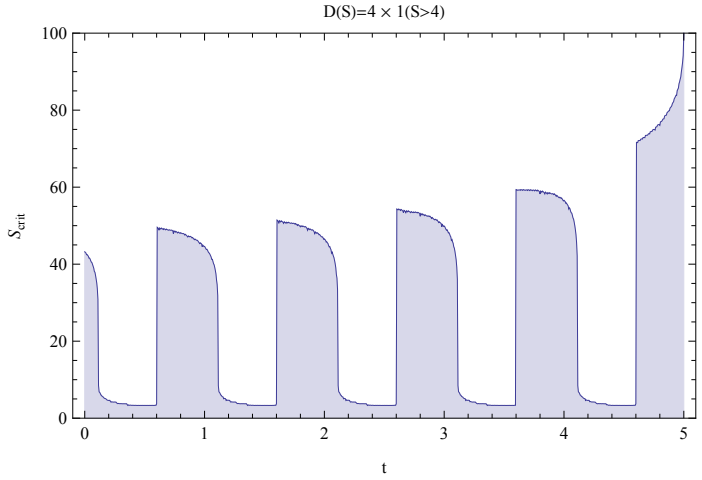
```
    Clear[CombinedChart];
    CombinedChart = Show[Vpluschart, Vminuschart,
      Graphics[point1], Graphics[point2], ImageSize → size];

    critdata = Table[{t1, GenericScrit[t1]}, {t1, 0, T, T / 1000}];
    p = ListPlot[critdata, PlotRange → {0, K},
      Filling → Axis, Joined → True, ImageSize → size,
            FrameLabel → {{"S_crit", Null}, {"t", label}} , Frame -> True];

    Return[GraphicsColumn[{p, CombinedChart}]];
   ] /; Td1 < T



(* Return interpolated value of x at first zero crossing pt *)
Clear[GenericXcritSearch];
GenericXcritSearch[xgrid_, t1_] :=
 Module[{tvfunc, len, xmin, f1, f2, m, x0, x1, x2, xcrit, dx},
  len = Length[xgrid];
  dx = 1.0 / len;
  xmin = xgrid[[2]];
  xcrit = xgrid[[1]];
  tvfunc[xx_] := GenericTV[xx, t1];
  f1 = tvfunc[xmin];
  If[f1 > 0, Return[xcrit], Null];

  x1 = xmin;
  For[x2 = xmin + dx, x2 < 0.5, x2 += dx,
   f2 = Chop[tvfunc[x2]];
   m = (f2 - f1) / dx;
   If[f2 > 0, xcrit = -f1 / m + x1; Break[], Null];
   x1 = x2; f1 = f2;
    ];
  Return[xcrit]]



(* HHL Liquidator policy with Div = 4, (td1,tdint) = (0.4,1) *)
PutDivPolicy[100, 100, 5, 0.08, 0.40, 1, 20000, 1, 250, 250, 0.4, 1, 4, 5, 400]
```

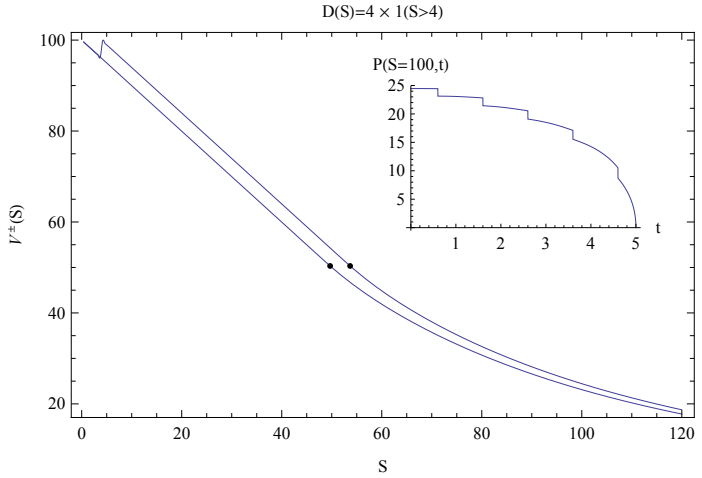
```
Solver: using div policy D(S)=Min[4, S] td1=0.4 tdint=1 T=5

Div detected at τ=0.4

Div detected at τ=1.4

Div detected at τ=2.4

Div detected at τ=3.4

Div detected at τ=4.4

solverDivPolicy: done; T=5 found 1250
  early exercise events including 5 discrete dividend events

PutDivPolicy:ct=13795 x0=0.5 npts=250 noppsperyear=250 Price=24.4582 (MMU=0.194909 GB)

At t=0 found Scrit=43.2251

At t=0.5996 with dx=0.004 found Scrit=0.

Creating Vpluschart at t=0.5996

Creating Vminuschart at t=0.6004

At t=0.6004 with dx=0.004 Scrit=49.6711
```





```
(* HHL Survivor policy with Div = 4, (td1,tdint) = (0.4,1) *)
PutDivPolicy[100, 100, 5, 0.08, 0.40, 1, 20 000, 1, 250, 250, 0.4, 1, 4, 5, 400]
Solver: using div policy D(S)=If[S < 4, 0, 4] td1=0.4 tdint=1 T=5

Div detected at τ=0.4

Div detected at τ=1.4

Div detected at τ=2.4

Div detected at τ=3.4

Div detected at τ=4.4

solverDivPolicy: done; T=5 found 1250
  early exercise events including 5 discrete dividend events

PutDivPolicy:ct=13 795 x0=0.5 npts=250 noppsperyear=250 Price=24.4582 (MMU=0.316435 GB)

At t=0 found Scrit=43.2251

At t=0.5996 with dx=0.004 found Scrit=3.73689

Creating Vpluschart at t=0.5996

Creating Vminuschart at t=0.6004

At t=0.6004 with dx=0.004 Scrit=49.6711
```






```
(* Book example: HHL Liquidator with Div = 4, (td1,tdint) = (0.4,1) *)
PutDivPolicy[100, 100, 5, 0.08, 0.40, 0, 20000, 1, 250, 250, 0.4, 1, 4, 5, 400]
```

```
Solver: using div policy D(S)=Min[4, S] td1=0.4 tdint=1 T=5

Div detected at τ=0.4

Div detected at τ=1.4

Div detected at τ=2.4

Div detected at τ=3.4

Div detected at τ=4.4

solverDivPolicy: done; T=5 found 1250
  early exercise events including 5 discrete dividend events

PutDivPolicy:ct=13795 x0=0.5 npts=250 noppsperyear=250 Price=24.4582 (MMU=0.711901 GB)

24.4582
```

```
PutDivPolicy[100, 100, 5, 0.08, 0.40, 0, 20000, 1, 250, 500, 0.4, 1, 4, 5, 400]
```

```
Solver: using div policy D(S)=Min[4, S] td1=0.4 tdint=1 T=5

Div detected at τ=0.4

Div detected at τ=1.4

Div detected at τ=2.4

Div detected at τ=3.4

solverEvent: ct=20 000 τ=3.6202 MIU(GB)=0.11228

Div detected at τ=4.4

solverDivPolicy: done; T=5 found 2500
  early exercise events including 5 discrete dividend events

PutDivPolicy:ct=27 682 x0=0.5 npts=250 noppsperyear=500 Price=24.4631 (MMU=0.367804 GB)

24.4631
```