```
   Help
#include  "bs1d_limdisc.h"
#include "pnl/pnl_cdf.h"

static int FF_call_down_out(double matu,double cont,double
    k,double valini,double r,double v,double bar_inf,double *pt
    _price,double *pt_delta)
{
  double  BB,AA,RR,CC,DD,EE,price1,delta1;

  BB=(log(k/valini)+((SQR(v))/2.-r)*matu)/(v*sqrt(matu));
  AA=(log(bar_inf/valini)+((SQR(v))/2.-r)*cont)/(v*sqrt(
    cont));
  RR=sqrt(cont/matu);
  CC=valini*v*sqrt(2.*M_PI*cont);
  DD=(log(valini/bar_inf)+(r+(SQR(v))/2.)*cont)/(v*sqrt(
    cont));
  EE=(log(valini/k)+(r+(SQR(v))/2.)*matu)/(v*sqrt(matu));

  pnl_cf_call_bs(bar_inf,k,matu-cont,r,0,v,&price1,&delta1)
    ;


  *pt_price=valini*pnl_cdf2nor(v*sqrt(cont)-AA,v*sqrt(matu)
    -BB,RR)-k*exp(-r*matu)*pnl_cdf2nor(-AA,-BB,RR);

  *pt_delta=exp(-(r*cont)-(0.5)*SQR(log(bar_inf/valini)-(r-
    SQR(v)/2.)*cont)/(cont*SQR(v)));

  *pt_delta=(*pt_delta)*(price1/CC)+ pnl_cdf2nor(DD,EE,RR);

  return OK;
}

static int Integration_call_down_out_BGK(double matu,
    double k,double r,double v,double bar_inf,int nb_bar,double u,
    double spot_en_u,double *pt_price,double *pt_delta)
{
  double a,b,beta,h,lambda,price_1,price_2,price_3,delta_1,
    delta_2,delta_3,
    spot1_en_u,c,der_c;
```

```
  int j;

  a=matu/(double)nb_bar;
  beta=0.5826;
  h=bar_inf*exp(-2*beta*v*sqrt(a));
  lambda=(2*r/SQR(v))-1;
  spot1_en_u=SQR(bar_inf)*exp(-2*beta*v*sqrt(a))/spot_en_u;
  c=pow((bar_inf*exp(-beta*v*sqrt(a)))/spot_en_u,lambda);
  der_c=-(lambda/bar_inf*exp(-beta*v*sqrt(a)))*pow(bar_inf*
    exp(-beta*v*sqrt(a))/spot_en_u,lambda+1);

  j=1;
  while(j<=(int)(u/a))
    j=j+1;

  b=(double)j*a;
  if(b==u)
    b=u+a;
  FF_call_down_out(matu-u,b-u,k,spot_en_u,r,v,bar_inf,&
    price_1,&delta_1);
  FF_call_down_out(matu-u,b-u,k,spot1_en_u,r,v,h,&price_2,&
    delta_2);
  pnl_cf_call_bs(spot1_en_u,k,matu-u,r,0.,v,&price_3,&delt
    a_3);
  *pt_price=price_1-c*price_3+c*price_2;
  *pt_delta=delta_1 - der_c*price_3 + c*delta_3*(spot1_en_
    u/spot_en_u) + der_c*price_2 - c*delta_2*(spot1_en_u/spot_
    en_u);

  return OK;
}

int CALC(AP_BroadieGlassermanKou)(void*Opt,void *Mod,Prici
    ngMethod *Met)
{
  TYPEOPT* ptOpt=( TYPEOPT*)Opt;
  TYPEMOD* ptMod=( TYPEMOD*)Mod;
  double r,limit,sd;
  int return_value;
```

```
  r=log(1.+ptMod->R.Val.V_DOUBLE/100.);
  limit=((ptOpt->Limit.Val.V_NUMFUNC_1)->Compute)((ptOpt->    Limit.Val.V_NUMFUN
  sd=(ptOpt->Limit.Val.V_NUMFUNC_1)->Par[0].Val.V_DATE;

  if(ptMod->Divid.Val.V_DOUBLE>0)
    {
      Fprintf(TOSCREEN,"Divid >0 , untreated case{n{n{n");
      return_value = WRONG;
    }
  else
    return_value=Integration_call_down_out_BGK(ptOpt->Matu
    rity.Val.V_DATE-sd,
                  (ptOpt->PayOff.Val.V_NUMFUNC_1)->Par[
    0].Val.V_PDOUBLE,
                  r,ptMod->Sigma.Val.V_PDOUBLE,limit,
                  (ptOpt->Limit.Val.V_NUMFUNC_1)->Par[2
    ].Val.V_INT2,
                  ptMod->T.Val.V_DATE-sd,
                  ptMod->S0.Val.V_PDOUBLE,
                  &(Met->Res[0].Val.V_DOUBLE),&(Met->
    Res[1].Val.V_DOUBLE));

  return return_value;

}

static int CHK_OPT(AP_BroadieGlassermanKou)(void *Opt, voi
    d *Mod)
{
  return strcmp( ((Option*)Opt)->Name,"CallDownOutDiscEuro"
    );
}

static int MET(Init)(PricingMethod *Met,Option *Opt)
{
  if ( Met->init == 0)
    {
      Met->init=1;
    }

  return OK;
```

```
}

PricingMethod MET(AP_BroadieGlassermanKou)=
{
  "AP_BroadieGlassermanKou",
  {{" ",PREMIA_NULLTYPE,{0},FORBID}},
  CALC(AP_BroadieGlassermanKou),
  {{"Price",DOUBLE,{100},FORBID},{"Delta",DOUBLE,{100},FORB
    ID} ,{" ",PREMIA_NULLTYPE,{0},FORBID}},
  CHK_OPT(AP_BroadieGlassermanKou),
  CHK_ok,
  MET(Init)
} ;
```

References