```
   Help
#include  "bs1d_limdisc.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.*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));

  Call_BlackScholes_73(bar_inf,k,matu-cont,r,0,v,
    &price1,&delta1);


  *pt_price=valini*NN(v*sqrt(cont)-AA,v*sqrt(matu
    )-BB,RR)-k*exp(-r*matu)*NN(-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)+ NN(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_pric
    e,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)))*po
    w(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,ba
    r_inf,&price_1,&delta_1);
  FF_call_down_out(matu-u,b-u,k,spot1_en_u,r,v,h,
    &price_2,&delta_2);
  Call_BlackScholes_73(spot1_en_u,k,matu-u,r,0.,
    v,&price_3,&delta_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,PricingMethod *Met)
```

```
{
  TYPEOPT* ptOpt=( TYPEOPT*)Opt;
  TYPEMOD* ptMod=( TYPEMOD*)Mod;
  double r,divid,limit,h,sd;
  int return_value;

  r=log(1.+ptMod->R.Val.V_DOUBLE/100.);
  divid=log(1.+ptMod->Divid.Val.V_DOUBLE/100.);
  limit=((ptOpt->Limit.Val.V_NUMFUNC_1)->Compute)
    ((ptOpt->Limit.Val.V_NUMFUNC_1)->Par,ptMod->T.
    Val.V_DATE);
  h=(ptOpt->Maturity.Val.V_DATE-(ptOpt->Limit.Val
    .V_NUMFUNC_1)->Par[0].Val.V_DATE)/(double)((pt
    Opt->Limit.Val.V_NUMFUNC_1)->Par[2].Val.V_INT2);
  sd=(ptOpt->Limit.Val.V_NUMFUNC_1)->Par[0].Val.
    V_DATE;

  if(ptMod->Divid.Val.V_DOUBLE>0)
    {
      Fprintf(TOSCREEN,"Divid >0 , untreated cas
    e{n{n{n");
      return_value = WRONG;
    }
  else
  return_value=Integration_call_down_out_BGK(pt
    Opt->Maturity.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_DA
    TE-sd,
                      ptMod->S0.Val.V_PD
    OUBLE,
                      &(Met->Res[0].Val.
    V_DOUBLE),&(Met->Res[1].Val.V_DOUBLE));

  return return_value;
```

```
}

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

static int MET(Init)(PricingMethod *Met)
{
  return OK;
}

PricingMethod MET(AP_BroadieGlassermanKou)=
{
  "AP_BroadieGlassermanKou",
  {{" ",END,0,FORBID}},
  CALC(AP_BroadieGlassermanKou),
  {{"Price",DOUBLE,100,FORBID},{"Delta",DOUBLE,10
    0,FORBID} ,{" ",END,0,FORBID}},
  CHK_OPT(AP_BroadieGlassermanKou),
  CHK_ok,
  MET(Init)
} ;
```

References