```
Help
#include "hullwhite2d stdi.h"
#include "hullwhite2d_includes.h"
#include "pnl/pnl_cdf.h"
//The "#else" part of the code will be freely available aft
    er the (year of creation of this file + 2)
#if defined(PremiaCurrentVersion) && PremiaCurrentVersion <</pre>
     (2009+2)
int CALC(CF_FLOORHW2D)(void *Opt,void *Mod,PricingMethod *
{
return AVAILABLE_IN_FULL_PREMIA;
static int CHK_OPT(CF_FLOORHW2D)(void *Opt, void *Mod)
  return NONACTIVE;
}
#else
// Volatility of an european option on a ZC bond P(T,S)
static double cf_ZBOvolatility2d(double a,double sigma1,
    double b, double sigma2, double rho, double t, double T, double S)
{
    double sigma_p;
    //double U, V, B TS;
    double exp_atT, exp_btT, exp_aTS, exp_bTS;
    double sigma3, eta, rhoG2;
    sigma3 = sqrt(sigma1*sigma1 + sigma2*sigma2/((b-a)*(b-
    a)) + 2*rho*sigma1*sigma2/(b-a));
    eta = sigma2 / (a-b);
    rhoG2 = (sigma1*rho - eta)/sigma3 ;
    exp atT = exp(-a*(T-t));
    exp btT = exp(-b*(T-t));
    exp_aTS = exp(-a*(S-T));
    exp bTS = exp(-b*(S-T));
    /* B_TS = (1 - exp_aTS) / a; */
```

```
/* U = (exp aTS - 1) * exp atT/(a*(a-b)); //(1/exp aS -
    1/\exp aT)/(a*(a-b)); */
    /* V = (exp_bTS - 1) * exp_btT/(b*(a-b)); // (1/exp_bS)
    - 1/\exp bT)/(b*(a-b)); */
    sigma p = sigma3*sigma3*(1-exp aTS)*(1-exp aTS)*(1-exp
    _atT*exp_atT)/(2*a*a*a) ;
    sigma_p += eta*eta*(1-exp_bTS)*(1-exp_bTS)*(1-exp_btT*
    \exp_btT)/(2*b*b*b);
    sigma p += 2*rhoG2*sigma3*eta*(1-exp aTS)*(1-exp bTS)*(
    1-\exp_atT*\exp_btT)/(a*b*(a+b));
    sigma_p = sqrt(sigma_p);
   return sigma p;
}
static double cf zbcall2d(ZCMarketData* ZCMarket,double t,
    double r, double u, double a, double sigma1, double b, double sigma2,
    double rho, double S, double T, double X)
{
    double PtS, PtT;
    double h, sigma p;
    double price;
    sigma_p = cf_ZBOvolatility2d( a, sigma1, b, sigma2, rh
    o, t, T, S);
   PtT=cf hw2d zcb(ZCMarket, a, sigma1, b, sigma2, rho, t,
    r, u, T);
    PtS=cf hw2d zcb(ZCMarket, a, sigma1, b, sigma2, rho, t,
    r, u, S);
   h = log(PtS/(PtT*X)) / sigma_p + 0.5 * sigma_p ;
    price = PtS * cdf_nor(h) - X * PtT * cdf_nor(h-sigma_p)
    ;
```

```
return price;
}
static int cf_floor2d(int flat_flag,double r_t,double u_t,
    double a, double sigma1, double b, double sigma2, double rho,
                            double Nominal, double K, double
    periodicity,double first_payement,double contract_maturity,
    double *price)
{
  double sum,tim,tip, strike_put;
  int i, nb_payement;
  ZCMarketData ZCMarket;
  /* Flag to decide to read or not ZC bond datas in "initia
    lyields.dat" */
  /* If P(0,T) not read then P(0,T)=\exp(-r0*T) */
  if(flat flag==0)
  {
      ZCMarket.FlatOrMarket = 0;
      ZCMarket.Rate = r_t;
  }
  else
  {
      ZCMarket.FlatOrMarket = 1;
      ReadMarketData(&ZCMarket);
      if(contract_maturity > GET(ZCMarket.tm,ZCMarket.Nvalu
    e-1))
      {
          printf("{nError : time bigger than the last time
    value entered in initialyield.dat{n");
          exit(EXIT FAILURE);
      }
  }
  strike_put = 1./(1 + periodicity*K);
```

```
nb payement=(int)((contract maturity-first payement)/pe
    riodicity);
  /*Cap=Portfolio of zero-bond Put options*/
  sum=0.;
  for(i=0; i<nb payement; i++)</pre>
  {
            = first payement + (double)i*periodicity;
      tip = tim + periodicity;
      sum += cf_zbcall2d(&ZCMarket, 0, r_t, u_t, a, sigma1
    , b, sigma2, rho, tip, tim, strike_put);
  sum = Nominal*(1.+K*periodicity)*sum;
  /*Price*/
  *price = sum;
  DeleteZCMarketData(&ZCMarket);
  return OK;
}
int CALC(CF FLOORHW2D)(void *Opt,void *Mod,PricingMethod *
    Met)
{
  TYPEOPT* ptOpt=(TYPEOPT*)Opt;
  TYPEMOD* ptMod=(TYPEMOD*)Mod;
  return cf_floor2d(
                        ptMod->flat_flag.Val.V_INT,
                        MOD(GetYield)(ptMod),
                        ptMod->InitialYieldsu.Val.V PDOUB
    LE,
                        ptMod->aR.Val.V_DOUBLE,
                        ptMod->SigmaR.Val.V PDOUBLE,
                        ptMod->bu.Val.V DOUBLE,
                        ptMod->Sigmau.Val.V_PDOUBLE,
                        ptMod->Rho.Val.V_PDOUBLE,
                        ptOpt->Nominal.Val.V PDOUBLE,
                        ptOpt->FixedRate.Val.V_PDOUBLE,
                        ptOpt->ResetPeriod.Val.V_DATE,
```

```
ptOpt->FirstResetDate.Val.V DATE-pt
    Mod->T.Val.V_DATE,
                        ptOpt->BMaturity.Val.V_DATE-ptMod->
    T.Val.V_DATE,
                        &(Met->Res[0].Val.V DOUBLE));
}
static int CHK_OPT(CF_FLOORHW2D)(void *Opt, void *Mod)
  return strcmp( ((Option*)Opt)->Name, "Floor");
#endif //PremiaCurrentVersion
static int MET(Init)(PricingMethod *Met,Option *Opt)
  if (Met->init == 0)
    {
      Met->init=1;
  return OK;
}
PricingMethod MET(CF FLOORHW2D)=
  "CF FloorHW2D",
  {{" ",PREMIA NULLTYPE,{0},FORBID}}},
  CALC(CF FLOORHW2D),
  {{"Price",DOUBLE,{100},FORBID}/*,{"Delta",DOUBLE,{100},FO
    RBID\ */,{" ",PREMIA_NULLTYPE,{0},FORBID}},
  CHK_OPT(CF_FLOORHW2D),
  CHK ok,
  MET(Init)
} ;
```

References