Help

```
#include  "hes1d_std.h"

#if defined(PremiaCurrentVersion) && PremiaCurrentVersion <
     (2011+2) //The "#else" part of the code will be freely av
    ailable after the (year of creation of this file + 2)
static int CHK_OPT(AP_AntonelliScarlatti_Heston)(void *Opt,
     void *Mod)
{
  return NONACTIVE;
}
int CALC(AP_AntonelliScarlatti_Heston)(void*Opt,void *Mod,
    PricingMethod *Met)
{
  return AVAILABLE_IN_FULL_PREMIA;
}
#else
////////////////////////////////////////////////////////////
    //////

// Computation of  d1
static double D1 ( double t, double x, double et, double T,
     double K, double r,double divid)
{
  return( (x-K+(r-divid)*(T-t)+1./2*(et*et))/et);
}
// Computation of s2
static  double D2 ( double t, double x, double et, double
    T, double K, double r,double divid)
{
  double d2;
  d2= D1(t,x,et,T,K,r,divid) - et;
  return ( d2);
}
// Variance of <M(t,v)>T in Heston model
static double variance_heston ( double t, double x, double
    v, double T,double a, double b, double c, double r, double
    K, double rho) {
  double va;
  va=  c * c * ( (2 * v) -  (2 * a * b * t) + 0.4e1 *  a *
    exp(- (b * (T - t))) +  (2 * a * b * T) -  (5 * a) - 0.2e1 *
      v * exp(- (2 * b * (T - t))) +  a * exp(- (2 * b * (T -
    t))) - 0.4e1 *  a * exp(- (b * (T - t))) *  b *  t + 0.4e1
    *  v * exp(- (b * (T - t))) *  b *  t + 0.4e1 *  a * exp(-
     (b * (T - t))) *  b *  T - 0.4e1 *  v * exp(- (b * (T -
    t))) *  b *  T) *  pow( b, (-3.)) / 0.2e1;

  return(va);
}

// Term g1 in Heston model
static double g1 (double t, double x, double v, double T,
    double a, double b, double c, double r, double divid,double K,
    double rho)
{

  double EM, sig,d2,G1;

  EM= a * (T - t) + (v - a) * (0.1e1 - exp(-b * (T - t))) /
     b; // expected value of <M(t,v)>T
  sig=sqrt(EM);
  d2=D2(t,x,sig,T,K,r,divid);
  G1= -((c*exp(K-r*(T-t))*d2*pnl_normal_density(d2))/(2*b*
    EM))*((v-2*a)/(b)*(1.-exp(-b*(T-t)))+(T-t)*(a-(v-a)*exp(-b*
    (T-t))));

  return(G1);

}

// Term g0 in Heston model
static double g0 (double t, double x, double v, double T,
    double a, double b, double c, double r, double divid,double K,
    double rho)
{
  double EM, sig,d1,d2,var,L;
  //expected value of <M(t,v)>T
  EM= a * (T - t) + (v - a) * (0.1e1 - exp(-b * (T - t))) /
     b;
  sig=sqrt(EM);
  d1=D1(t,x,sig,T,K,r,divid);
  d2=D2(t,x,sig,T,K,r,divid);

  var= variance_heston(t,x,v,T,a,b,c,r,K,rho);

  L= exp(x) * cdf_nor(d1) -  exp(K - r * (T - t)) * cdf_nor
    (d2) + exp(K - r * (T - t)) *  (d2 * d1 - 1) * pnl_normal_
    density(d2) * var *  pow( EM,  -0.3e1 / 0.2e1) / 0.8e1;

  return(L);

}


//a long term run
//b speed of mean reversion
//c vol of vol
int ApAntonelliScarlattiHeston(double S,NumFunc_1  *p,
    double T, double r, double divid1, double v,double b,double a,
    double c,double rho,double *ptprice, double *ptdelta)
{
  int flag_call;
  double K,x,price,delta;
  double g,gh;
  double EM,d1,d2,Ec,h0,h1,t;
  double divid;

  K=p->Par[0].Val.V_PDOUBLE;
  divid=0;
  r=r-divid1;
  //Log trasformation
  K=log(K);
  x=log(S);
  t=0.;

  if ((p->Compute)==&Call)
    flag_call=1;
  else
    flag_call=0;

  //Pricing
  g=g0(t,x,v,T,a,b,c,r,divid,K,rho);
  gh=g1(t,x,v,T,a,b,c,r,divid,K,rho);

  //Hedging
  EM= a*T+(v-a)/b*(1.-exp(-b*T)) ;
  d1=D1(t,x,sqrt(EM),T,K,r,divid);
  d2=D2(t,x,sqrt(EM),T,K,r,divid);
  Ec=c * (a / b * T - (v - a) / b * T * exp(-b * T) + (v -
    0.2e1 * a) * pow(b, -0.2e1) * (0.1e1 - exp(-b * T))) / 0.2
    e1;

  // h0 term //
  h0=exp(x)*cdf_nor(d1);

  // h1 term //
  h1= -exp(K-r*T)*(1.-(d2)*(d2))*pnl_normal_density(d2)*Ec/
    (pow(EM,1.5));

  //Call case
  if(flag_call==1)
    {
      price=g+rho*gh;
      delta=(h0+rho*h1)*exp(-x);
    }//Put case
  else
    {
      price=g+rho*gh-S+exp(K-r*T);
      delta=(h0+rho*h1)*exp(-x)-1.;
    }

  /* Price*/
  *ptprice=price*exp(-divid1*T);

  /* Delta */
  *ptdelta=delta*exp(-divid1*T);

  return OK;
}

int CALC(AP_AntonelliScarlatti_Heston)(void *Opt, void *
    Mod, PricingMethod *Met)
```

```
{
  TYPEOPT* ptOpt=(TYPEOPT*)Opt;
  TYPEMOD* ptMod=(TYPEMOD*)Mod;
  double r,divid;

  if(ptMod->Sigma.Val.V_PDOUBLE==0.0)
    {
      Fprintf(TOSCREEN,"BLACK-SHOLES MODEL{n{n{n");
      return WRONG;
    }
  else
    {
      r=log(1.+ptMod->R.Val.V_DOUBLE/100.);
      divid=log(1.+ptMod->Divid.Val.V_DOUBLE/100.);

      return ApAntonelliScarlattiHeston(ptMod->S0.Val.V_PDO
    UBLE,
                                        ptOpt->PayOff.Val.
    V_NUMFUNC_1,
                                        ptOpt->Maturity.Val
    .V_DATE-ptMod->T.Val.V_DATE,
                                        r,
                                        divid, ptMod->Sigma
    0.Val.V_PDOUBLE
                                        ,ptMod->MeanReversion.h
    al.V_PDOUBLE,
                                        ptMod->LongRunVaria
    nce.Val.V_PDOUBLE,
                                        ptMod->Sigma.Val.V_
    PDOUBLE,
                                        ptMod->Rho.Val.V_
    PDOUBLE,
                                        &(Met->Res[0].Val.
    V_DOUBLE),
                                        &(Met->Res[1].Val.
    V_DOUBLE)
        );
    }

}
```

```
static int CHK_OPT(AP_AntonelliScarlatti_Heston)(void *Opt,
     void *Mod)
{
  if ((strcmp( ((Option*)Opt)->Name,"CallEuro")==0)
      ||(strcmp( ((Option*)Opt)->Name,"PutEuro")==0))

    return OK;
  return WRONG;
}

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

  return OK;
}

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

References