Help

```
extern"C" {
#include "
href../../../mod/doublehes1d/doublehes1d_vol/doublehes1d_vol_h_src.pdfhes1d_vol.
#include "
href../../../common/numfunc_h_src.pdfnumfunc.h"
}

#include "
href../../../common/math/intg_h_src.pdfmath/intg.h"

extern "C" {




#if defined(PremiaCurrentVersion) && PremiaCurrentVersion < (2008+2) //The "#els
  static int CHK_OPT(AP_HES_VOLATILITYSWAP)(void *Opt, void *Mod)
  {
    return NONACTIVE;
  }
  int CALC(AP_HES_VOLATILITYSWAP)(void *Opt, void *Mod, PricingMethod *Met)
  {
    return AVAILABLE_IN_FULL_PREMIA;
  }
#else

  static double v0, kk, tet, sgm, tt;

  static double Phi(double x)
  {
    double d, edt, ss, divedt, aa, bb, val;

    ss = sgm * sgm;
    d = sqrt(kk * kk + 2.0 * ss * x);
    edt = exp(-d * tt);
    divedt = 1.0 + kk / d + (1.0 - kk / d) * edt;
    aa = 2.0 * tet * kk / ss * ((kk - d) * tt / 2.0 + log(2.0 / divedt));
    bb = -v0 * x / d * 2.0 * (1.0 - edt) / divedt;
```

```
  val = exp(aa + bb);

  return val;
}
/*////////////////////////////////////////////////////*/
static double funct(double x)
{
  if (x == 0)
    {
      return 1.0;
    }
  else
    {
      return 1.0 - Phi(1.0 / x / x);
    }
}

/*////////////////////////////////////////////////////*/
static double intLvar(double Lam)
{
  double res, ae, temp;
  int i;
  //  intg(0.0, Lam, funct, 1e-14, 1e-10, &res, &ae);

  temp = 0.0;
  Lam = 2.0 * Lam / 100.0;
  intg(0.0, Lam, funct, 1e-14, 1e-10, &res, &ae);
  temp += res;
  for (i = 1; i < 101; i++)
    {
      intg(i * Lam, (i + 1)*Lam, funct, 1e-14, 1e-10, &res, &ae);
      temp += res;

    }
  res = temp;

  return res;
}
/*////////////////////////////////////////////////////*/
static int ap_hes_volswap(double sigma0, double ka, double theta, double sigma
                          double r, double divid, double T, double Strike,
```

```
                              double Spot, double *fairval, double *Price)
{
  double int_oe, int_ei;
  double eVar, eVol, ekt;
  double eps = 1.0e-6;

  kk = ka;
  ka *= T;
  ekt = exp(-ka);
  eVar = theta + (sigma0 - theta) * (1.0 - ekt) / ka;

  //approximation with Laplace-----------------------------------------
  v0 = sigma0;
  tet = theta;
  sgm = sigma2;
  tt = T;

  int_oe = 2.0 * eVar * sqrt(eps); // =int_0^eps

  int_ei = 2.0 * intLvar(1.0 / sqrt(eps)); // =int_eps^inf

  eVol = (int_oe + int_ei) * 0.5 / sqrt(M_PI) / sqrt(tt);
  //fair strike of volatility swap
  *fairval = eVol * 100;
  // price of vol swap
  *Price =  exp(-r * T) * (*fairval - Strike);

  return OK;
}

/*------------------------------------------------------------*/

int CALC(AP_HES_VOLATILITYSWAP)(void *Opt, void *Mod, PricingMethod *Met)
{
  TYPEOPT *ptOpt = (TYPEOPT *)Opt;
  TYPEMOD *ptMod = (TYPEMOD *)Mod;
  double r, divid, strike, spot;
  NumFunc_1 *p;

  r = log(1. + ptMod->R.Val.V_DOUBLE / 100.);
  divid = log(1. + ptMod->Divid.Val.V_DOUBLE / 100.);
```

```
    p = ptOpt->PayOff.Val.V_NUMFUNC_1;
    strike = p->Par[0].Val.V_DOUBLE;
    spot = ptMod->S0.Val.V_DOUBLE;

    return ap_hes_volswap(
             ptMod->Sigma0.Val.V_PDOUBLE
             , ptMod->MeanReversion.Val.V_PDOUBLE,
             ptMod->LongRunVariance.Val.V_PDOUBLE,
             ptMod->Sigma.Val.V_PDOUBLE,
             ptMod->Rho.Val.V_PDOUBLE,
             r, divid,
             ptOpt->Maturity.Val.V_DATE - ptMod->T.Val.V_DATE,
             strike, spot,
             &(Met->Res[0].Val.V_DOUBLE)/*FAIRVAL*/,
             &(Met->Res[1].Val.V_DOUBLE)/*PRICE*/);

  }

  static int CHK_OPT(AP_HES_VOLATILITYSWAP)(void *Opt, void *Mod)
  {
    if ((strcmp(((Option *)Opt)->Name, "VolatilitySwap") == 0))
      return OK;

    return WRONG;
  }

#endif //PremiaCurrentVersion
  static int MET(Init)(PricingMethod *Met, Option *Opt)
  {

    return OK;
  }

  PricingMethod MET(AP_HES_VOLATILITYSWAP) =
  {
    "AP_HES_VOLATILITYSWAP", //"Integral representation",
    {   {" ", PREMIA_NULLTYPE, {0}, FORBID}},
    CALC(AP_HES_VOLATILITYSWAP),
    { {"Fair strike in annual volatility points", DOUBLE, {100}, FORBID},
      {"Price", DOUBLE, {100}, FORBID},
      {" ", PREMIA_NULLTYPE, {0}, FORBID}
```

```
    },
    CHK_OPT(AP_HES_VOLATILITYSWAP),
    CHK_ok ,
    MET(Init)
  } ;

  /*////////////////////////////////////*/
}
```