Help

```
#include  "variancegamma2d_std2dg.h"
#include"pnl/pnl_complex.h"
#include "pnl/pnl_cdf.h"


#if defined(PremiaCurrentVersion) && PremiaCurrentVersion < (2013+2) //The "#els
static int CHK_OPT(AP_Spread_HurdZhou_vg)(void *Opt, void *Mod)
{
  return NONACTIVE;
}
int CALC(AP_Spread_HurdZhou_vg)(void *Opt, void *Mod, PricingMethod *Met)
{
  return AVAILABLE_IN_FULL_PREMIA;
}
#else

static dcomplex charact_vg(dcomplex u1, dcomplex u2, double a_m, double a_p, dou
                           double lambda, double t)
{

  dcomplex aaa, s, ppp, fact_s, fact1, fact2, res;
  aaa.r = 0;
  aaa.i = 1 / a_m - 1 / a_p;

  s = Cadd(u1, u2);

  ppp.r = 1 / a_m / a_p;
  ppp.i = 0;

  fact_s = CRadd(Cmul(Cadd(aaa, Cmul(ppp, s)), s), 1.);

  fact1 = CRadd(Cmul(Cadd(aaa, Cmul(ppp, u1)), u1), 1.);

  fact2 = CRadd(Cmul(Cadd(aaa, Cmul(ppp, u2)), u2), 1.);

  res = Cmul(Cpow_real(fact_s, -alpha * lambda * t), Cpow_real(Cmul(fact1, fact2

  return res;
}
```

```
static dcomplex p_hat(dcomplex u1, dcomplex u2)
{
  dcomplex T1, T2, T3, I;
  I.r = 0.0;
  I.i = 1.0;
  T1 = Ctgamma(CRsub(Cmul(I, Cadd(u1, u2)), 1.0));
  T2 = Ctgamma(Cmul(Cminus(I), u2));
  T3 = Ctgamma(CRadd(Cmul(I, u1), 1.0));
  return Cmul(T1, Cdiv(T2, T3));
}

static dcomplex big_h_vg(double u1, double u2, double eps1, double eps2, double
                         double a_p, double alpha, double lambda, double t, doub
{
  dcomplex T1, T2, z1, z2, res, I;
  I.r = 0.0;
  I.i = 1.0;
  z1.r = u1;
  z1.i = eps1;
  z2.r = u2;
  z2.i = eps2;
  res = Cmul(p_hat(z1, z2), charact_vg(z1, z2, a_m, a_p, alpha, lambda, t));


  T1 = Cmul(I, RCmul(x1, z1));
  T2 = Cmul(I, RCmul(x2, z2));
  res = Cmul(res, Cexp(Cadd(T1, T2)));
  return res;
}

static int Spread_HurdZhou_vgAn(double s01, double s02, double K, double t, doub
{
  double u_hat, eps1, eps2;
  unsigned int N;
  double x1, x2;
  double eta, fact;
  unsigned long k1, k2;
  dcomplex z;
  double u1, u2, d11, d22;
  dcomplex res, dzdx1, dzdx2, I;
```

```
I.r = 0.0;
I.i = 1.0;

// 2D Intregration parameters
N = 256;
u_hat = 40;
eps1 = -5;
eps2 = 3;


if (K < 0)
  {
    double e1 = 1.0, e2 = 1.0;
    Spread_HurdZhou_vgAn(s02, s01, -K, t, r, a_m, a_p, alpha, lambda,
                         ptprice, &d22, &d11);
    d11 += e1;
    d22 -= e2;
    if (ptdelta1 != 0)
      {
        *ptdelta1 = d11;
      }
    if (ptdelta2 != 0)
      {
        *ptdelta2 = d22;
      }
    *ptprice = (*ptprice) - s02 * e2 + s01 * e1 - exp(-r * t) * K;
  }

else if (K == 0)
  {
    K = (s01 + s02) * 0.00001;
    Spread_HurdZhou_vgAn(s01 / K, s02 / K, 1, t, r, a_m, a_p, alpha, lambda,
                         ptprice, ptdelta1, ptdelta2);
    *ptprice = (*ptprice) * K;
  }

else if (K != 1)
  {
    Spread_HurdZhou_vgAn(s01 / K, s02 / K, 1, t, r, a_m, a_p, alpha, lambda,
                         ptprice, ptdelta1, ptdelta2);
    *ptprice = (*ptprice) * K;
```

```
  }
else if (K == 1)
  {
    x1 = log(s01);
    x2 = log(s02);

    eta = u_hat * 2 / N;

    res.r = 0.0;
    res.i = 0.0;
    dzdx1.r = 0.0;
    dzdx1.i = 0.0;
    dzdx2.r = 0.0;
    dzdx2.i = 0.0;

    for (k1 = 0; k1 < N; k1++)
      {
        u1 = -u_hat + k1 * eta;
        for (k2 = 0; k2 < N; k2++)
          {
            u2 = -u_hat + k2 * eta;
            z = big_h_vg(u1, u2, eps1, eps2, a_m, a_p, alpha, lambda, t, x1, x
            res = Cadd(res, z);
            dzdx1 = Cadd(dzdx1, Cmul(CRsub(CRmul(I, u1), eps1), z));
            dzdx2 = Cadd(dzdx2, Cmul(CRsub(CRmul(I, u2), eps2), z));
          }
      }

    fact = u_hat / N / M_PI;
    fact = fact * fact;
    fact *= exp(-r * t);
    res = CRmul(res, fact);

    if (ptdelta1 != 0)
      {
        *ptdelta1 = Creal(dzdx1) * fact / s01;
      }
    if (ptdelta2 != 0)
      {
        *ptdelta2 = Creal(dzdx2) * fact / s02;
      }
```

```
      *ptprice = Creal(res);
    }

  return OK;
}

int CALC(AP_Spread_HurdZhou_vg)(void *Opt, void *Mod, PricingMethod *Met)
{
  TYPEOPT *ptOpt = (TYPEOPT *)Opt;
  TYPEMOD *ptMod = (TYPEMOD *)Mod;
  double r;

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

  //CallSpread
  if ((ptOpt->PayOff.Val.V_NUMFUNC_2)->Compute == CallSpread2d)
    return Spread_HurdZhou_vgAn(ptMod->S01.Val.V_PDOUBLE, ptMod->S02.Val.V_PDOUB
  else//PutSpread with -strike
    return Spread_HurdZhou_vgAn(ptMod->S01.Val.V_PDOUBLE, ptMod->S02.Val.V_PDOUB
}

static int CHK_OPT(AP_Spread_HurdZhou_vg)(void *Opt, void *Mod)
{
  if ((strcmp(((Option *)Opt)->Name, "CallSpread2dEuro") == 0) || (strcmp(((Opti
    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_Spread_HurdZhou_vg) =
{
```

```
  "AP_Spread_HurdZhou_vg",
  {{" ", PREMIA_NULLTYPE, {0}, FORBID}},
  CALC(AP_Spread_HurdZhou_vg),
  { {"Price", DOUBLE, {100}, FORBID}, {"Delta1", DOUBLE, {100}, FORBID} , {"Delt
    {" ", PREMIA_NULLTYPE, {0}, FORBID}
  },
  CHK_OPT(AP_Spread_HurdZhou_vg),
  CHK_ok,
  MET(Init)
} ;
```