

[Help](#)

```

#include "hes1d_vol.h"
#include "numfunc.h"
#include "pnl/pnl_mathtools.h"

#if defined(PremiaCurrentVersion) && PremiaCurrentVersion < (2014+2) //The "#els
static int CHK_OPT(AP_HES_VS_ZHOU)(void *Opt, void *Mod)
{
    return NONACTIVE;
}
int CALC(AP_HES_VS_ZHOU)(void *Opt, void *Mod, PricingMethod *Met)
{
    return AVAILABLE_IN_FULL_PREMIA;
}
#else

static double C_tilde(double kappa, double theta, double sigma, double v0, doubl
{
    double a_tilde, b_tilde, g_tilde;

    a_tilde = kappa - 2.0 * rho * sigma;
    b_tilde = sqrt(SQR(a_tilde) - 2.0 * SQR(sigma));
    g_tilde = SQR(a_tilde / sigma) - 1.0 + a_tilde / sigma * sqrt(SQR(a_tilde / si

    return (tau * deltaT + kappa * theta / SQR(sigma) * ((a_tilde + b_tilde) * del
}

static double D_tilde(double kappa, double theta, double sigma, double v0, doubl
{
    double a_tilde, b_tilde, g_tilde;

    a_tilde = kappa - 2.0 * rho * sigma;
    b_tilde = sqrt(SQR(a_tilde) - 2.0 * SQR(sigma));
    g_tilde = SQR(a_tilde / sigma) - 1.0 + a_tilde / sigma * sqrt(SQR(a_tilde / si

    return ((a_tilde + b_tilde) / SQR(sigma) * (1.0 - exp(b_tilde * deltaT)) / (1
}

```

```

/*////////////////////////////////////*/
static int ap_hes_varswap_zhou(double v0, double kappa, double theta, double sig
{
    int N;
    int i;
    double t, c_i;
    double deltaT;
    double C, D;

    //Number of discrete times
    N = 252;

    deltaT = mat / (double)N;

    C = C_tilde(kappa, theta, sigma, v0, rho, r, deltaT);
    D = D_tilde(kappa, theta, sigma, v0, rho, r, deltaT);

    *ptprice = exp(C + D * v0) + exp(-r * deltaT) - 2.0;
    for (i = 2; i <= N; i++)
    {
        t = (i - 1.0) * mat / (double)N;
        c_i = 2.0 * kappa / (SQR(sigma) * (1.0 - exp(-kappa * t)));

        *ptprice += exp(C + c_i * exp(-kappa * t) / (c_i - D) * D * v0) * exp(2.0
    }
    *ptprice = *ptprice * 10000;
    *ptprice *= exp(r * deltaT) / mat;
    *fairval = *ptprice - SQR(strike);

    return OK;
}

int CALC(AP_HES_VS_ZHOU)(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_varswap_zhou(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), &(Met->Res[1].Val.V_DOUBLE)
                               );
}

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

    return WRONG;
}

#endif //PremiaCurrentVersion
static int MET(Init)(PricingMethod *Met, Option *Opt)
{
    Met->HelpFilenameHint = "ap_hes_zhou";
    return OK;
}

PricingMethod MET(AP_HES_VS_ZHOU) =
{
    "AP_HES_VS_ZHOU",
    { {" ", PREMIA_NULLTYPE, {0}, FORBID}},
    CALC(AP_HES_VS_ZHOU),
    { {"Price in 10000 variance points", DOUBLE, {100}, FORBID}, {"Fair strike for
      {" ", PREMIA_NULLTYPE, {0}, FORBID}

```

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