

[Help](#)

```

#include "merhes1d_vol.h"
#include "pnl/pnl_integration.h"

#if defined(PremiaCurrentVersion) && PremiaCurrentVersion < (2010+2) //The "#els
static int CHK_OPT(AP_MERHES_VOLATILITYSWAP)(void *Opt, void *Mod)
{
    return NONACTIVE;
}
int CALC(AP_MERHES_VOLATILITYSWAP)(void *Opt, void *Mod, PricingMethod *Met)
{
    return AVAILABLE_IN_FULL_PREMIA;
}
#else

static double v0, kk, tet, sgm, tt, gam, mu, del;

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

    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;
    // jumping part
    divedt = 2.0 * del * del * x + 1.0;
    cc = exp(-mu * mu * x / divedt) / sqrt(divedt) ;
    cc = gam * tt * (cc - 1.0);

    val = exp(aa + bb + cc);

    return val;
}
/*////////////////////////////////////*/
static double funct(double x, void *p)

```

```

{
    if (x == 0)
    {
        return 1.0;
    }
    else
    {
        return 1.0 - Phi(1.0 / x / x);
    }
}

/*////////////////////////////////////*/
static double intLvar(double Lam)
{
    double temp;
    int i;
    double result, abserr;
    int neval;

    PnlFunc func;
    func.F = funct;
    func.params = NULL;

    temp = 0.0;

    Lam = 2.0 * Lam / 100.0;
    pnl_integration_GK(&func, 0.0, Lam, 0.000001, 0.0000001, &result, &abserr, &neval);

    temp += result;
    for (i = 1; i < 101; i++)
    {
        pnl_integration_GK(&func, i * Lam, (i + 1)*Lam, 0.000001, 0.0000001, &result, &abserr, &neval);
        temp += result;
    }
    result = temp;

    return result;
}

/*////////////////////////////////////*/
static int ap_merhes_volswap(double sigma0, double ka, double theta, double sigma1)

```

```

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

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

//approximation with Laplace-----
v0 = sigma0;
tet = theta;
sgm = sigma2;
tt = T;
gam = gamma;
mu = nu;
del = delta;

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_MERHES_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_merhes_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,
        ptMod->Lambda.Val.V_PDOUBLE,
        ptMod->Mean.Val.V_DOUBLE,
        ptMod->Variance.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_MERHES_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_MERHES_VOLATILITYSWAP) =

```

```
{
  "AP_MERHES_VOLATILITYSWAP",
  { {" " , PREMIA_NULLTYPE, {0}, FORBID}},
  CALC(AP_MERHES_VOLATILITYSWAP),
  { {"Fair strike in annual volatility points", DOUBLE, {100}, FORBID},
    {"Price " , DOUBLE, {100}, FORBID},
    {" " , PREMIA_NULLTYPE, {0}, FORBID}
  },
  CHK_OPT(AP_MERHES_VOLATILITYSWAP),
  CHK_ok ,
  MET(Init)
} ;

/*////////////////////////////////////////*/
```