

[Help](#)

```
#include "
href../../../../mod/merhes1d/merhes1d_pad/merhes1d_pad_h_src.pdfhes1d_pad.h"
#include <pnl/pnl_mathtools.h>
#include <pnl/pnl_root.h>

#if defined(PremiaCurrentVersion) && PremiaCurrentVersion < (2012+2) //The "#els
static int CHK_OPT(AP_FJM_ASIAN_HESTON)(void *Opt, void *Mod)
{
    return NONACTIVE;
}
int CALC(AP_FJM_ASIAN_HESTON)(void *Opt, void *Mod, PricingMethod *Met)
{
    return AVAILABLE_IN_FULL_PREMIA;
}
#else

//Calculating the second moment of  $\int_0^T S_t dt$  under Heston model
static double secondMomentAsianHeston(double kappa, double theta, double sigma,
{
    int N = 10000;
    int i;
    double res = 0.0;
    double ES2 = 0.0;
    double u ;
    double gamma = sqrt(SQR(kappa - 2.0 * rho * sigma) - 2.0 * SQR(sigma));

    for (i = 0; i < N; i++)
    {
        u = T * i / (double)N ;
        ES2 += T / (double)N * exp(mu * u) * exp((kappa - 2.0 * rho * sigma) * kap
            cosh(0.5 * gamma * u) + (kappa - 2.0 * rho * sigma) / gamma * sin

        res += T / (double)N * exp(mu * u) * ES2;
    }
    return 2.0 * res * S0 * S0;
}

static int APAsianFJM(double S0, double K, double T, double r, double div, doubl
```

```

{
    double F, EI2, var;

    if (r - div != 0)
        F = S0 * (exp((r - div) * T) - 1.0) / (r - div);
    else
        F = S0 * T;

    EI2 = secondMomentAsianHeston(kappa, theta, sigma, rho, v0, S0, r - div, T);

    var = (log(EI2) - 2.0 * log(F)) / T;
    //mean = log(F)/T - 0.5*var ;

    pnl_cf_call_bs(F / T * exp(-r * T), K, T, r, 0.0, sqrt(var), price, delta);

    return OK;
}

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

    r = log(1. + ptMod->R.Val.V_DOUBLE / 100.);
    divid = log(1. + ptMod->Divid.Val.V_DOUBLE / 100.);
    time_spent = (ptMod->T.Val.V_DATE - (ptOpt->PathDep.Val.V_NUMFUNC_2)->Par[0].V
    pseudo_strike = (ptOpt->PayOff.Val.V_NUMFUNC_2)->Par[0].Val.V_PDOUBLE - time_s
    return APAsianFJM(ptMod->S0.Val.V_PDOUBLE,
        pseudo_strike,
        ptOpt->Maturity.Val.V_DATE - ptMod->T.Val.V_DATE,
        r,
        divid, 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,
        &(Met->Res[0].Val.V_DOUBLE),
        &(Met->Res[1].Val.V_DOUBLE)
    );
}

```

```

static int CHK_OPT(AP_FJM_ASIAN_HESTON)(void *Opt, void *Mod)
{
    if ((strcmp(((Option *)Opt)->Name, "AsianCallFixedEuro") == 0))
    {
        return OK;
    }
    return WRONG;
}

```

```

#endif //PremiaCurrentVersion

```

```

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

    return OK;
}

```

```

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

```