

[Help](#)

```
#include "
href../../../../mod/merhes1d/merhes1d_std/merhes1d_std_h_src.pdfmerhes1d_std.h"
#include "
href../../../../common/math/equity_pricer/levy_diffusion_h_src.pdfmath/equity_price
#include "
href../../../../common/math/equity_pricer/carr_h_src.pdfmath/equity_pricer/carr.h"

#if defined(PremiaCurrentVersion) && PremiaCurrentVersion < (2010+2) //The "#els
static int CHK_OPT(CF_AttariMertonHeston)(void *Opt, void *Mod)
{
    return NONACTIVE;
}
int CALC(CF_AttariMertonHeston)(void *Opt, void *Mod, PricingMethod *Met)
{
    return AVAILABLE_IN_FULL_PREMIA;
}
#else
int CALC(CF_AttariMertonHeston)(void *Opt, void *Mod, PricingMethod *Met)
{
    TYPEOPT *ptOpt = (TYPEOPT *)Opt;
    TYPEMOD *ptMod = (TYPEMOD *)Mod;
    NumFunc_1 *p;
    int option_type;
    int std = 1;
    if (ptMod->Sigma.Val.V_PDOUBLE == 0.0)
    {
        Fprintf(TOSCREEN, "BLACK-SHOLES MODEL\ n\ n\ n");
        return WRONG;
    }
    else
    {
        double drift;
        Option_Eqd *op;
        Bates_diffusion *Process = Bates_diffusion_create(ptMod->LongRunVariance.V
            ptMod->MeanReversion.Val.V_PDOUBLE,
            ptMod->Rho.Val.V_PDOUBLE,
            ptMod->Sigma.Val.V_PDOUBLE,
            sqrt(ptMod->Sigma0.Val.V_PDOUBLE),
            ptMod->Mean.Val.V_PDOUBLE,
```

```

                                sqrt(ptMod->Variance.Val.V_PDOUBLE),
                                ptMod->Lambda.Val.V_PDOUBLE,
                                &drift);
    Levy_diffusion *Levy = Levy_diffusion_create(Process, &Bates_diffusion_cha
    p = ptOpt->PayOff.Val.V_NUMFUNC_1;
    if ((p->Compute) == &Call)
        option_type = 1;
    else if ((p->Compute) == &Put)
        option_type = 2;
    else
        option_type = 3;

    op = option_eqd_create(ptOpt->EuOrAm.Val.V_BOOL, option_type, std, ptMod->
    option_eqd_set_rate(op, log(1. + ptMod->R.Val.V_DOUBLE / 100.), log(1. + p

    AttariMethod_Vanilla_option_LD(op, 0.1, Levy);
    (Met->Res[0].Val.V_DOUBLE) = op->price;
    (Met->Res[1].Val.V_DOUBLE) = op->delta;
    free(op);
    free(Levy);
    free(Process);
    return OK;
}
}

static int CHK_OPT(CF_AttariMertonHeston)(void *Opt, void *Mod)
{
    if ((strcmp(((Option *)Opt)->Name, "CallEuro") == 0) || (strcmp(((Option *)Opt
        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(CF_AttariMertonHeston) =
{
    "CF_Attari_MerHes",
    {{ " ", PREMIA_NULLTYPE, {0}, FORBID}},
    CALC(CF_AttariMertonHeston),
    { {"Price", DOUBLE, {100}, FORBID},
      {"Delta", DOUBLE, {100}, FORBID} ,
      {" ", PREMIA_NULLTYPE, {0}, FORBID}
    },
    CHK_OPT(CF_AttariMertonHeston),
    CHK_ok,
    MET(Init)
};

```