

[Help](#)

```
#include "
href../../../../mod/libor_affine_cir1d/libor_affine_cir1d_std/libor_affine_cir1d_s
#include "
href../../../../common/math/libor_affine_model/libor_affine_framework_h_src.pdfmath
#include "
href../../../../common/math/libor_affine_model/libor_affine_pricing_h_src.pdfmath/l
#include "
href../../../../common/math/libor_affine_model/libor_affine_models_h_src.pdfmath/li

#if defined(PremiaCurrentVersion) && PremiaCurrentVersion < (2011+2) //The "#els
static int CHK_OPT(CF_LibAffCir1d_Fourier_Swaption)(void *Opt, void *Mod)
{
    return NONACTIVE;
}
int CALC(CF_LibAffCir1d_Fourier_Swaption)(void *Opt, void *Mod, PricingMethod *M
{
    return AVAILABLE_IN_FULL_PREMIA;
}
#else

static int cf_swaption_fourier_libaff_cir1d(int InitYieldCurve_flag, double R_flat)
{
    StructLiborAffine LiborAffine;
    ZCMarketData ZCMarket;
    PnlVect *ModelParams = pnl_vect_create(4);

    ZCMarket.filename = curve;
    LET(ModelParams, 0) = x0;
    LET(ModelParams, 1) = lambda;
    LET(ModelParams, 2) = theta;
    LET(ModelParams, 3) = eta;

    SetInitYieldCurve(InitYieldCurve_flag, R_flat, &ZCMarket);

    CreateStructLiborAffine(&LiborAffine, &ZCMarket, swaption_start, swaption_end,

    *swaption_price = cf_swaption_fourier_libaff(&LiborAffine, swaption_start, swa
```

```

FreeStructLiborAffine(&LiborAffine);

return OK;
}

///  

//***** PREMIA FUNCTIONS *****  

int CALC(CF_LibAffCir1d_Fourier_Swaption)(void *Opt, void *Mod, PricingMethod *M  

{
    TYPEOPT *ptOpt = (TYPEOPT *)Opt;  

    TYPEMOD *ptMod = (TYPEMOD *)Mod;  

  

    int swaption_payer_receiver = (((ptOpt->PayOff.Val.V_NUMFUNC_1)->Compute) == &  

  

    return  cf_swaption_fourier_libaff_cir1d(ptMod->flat_flag.Val.V_INT,  

        MOD(GetYield)(ptMod),  

        MOD(GetCurve)(ptMod),  

        ptMod->x0.Val.V_DOUBLE,  

        ptMod->lambda.Val.V_PDOUBLE,  

        ptMod->theta.Val.V_DOUBLE,  

        ptMod->eta.Val.V_PDOUBLE,  

        ptOpt->OMaturity.Val.V_DATE - ptMod->T.Val.V_DATE,  

        ptOpt->BMaturity.Val.V_DATE - ptMod->T.Val.V_DATE,  

        ptOpt->ResetPeriod.Val.V_DATE,  

        ptOpt->FixedRate.Val.V_PDOUBLE,  

        ptOpt->Nominal.Val.V_PDOUBLE,  

        swaption_payer_receiver,  

        &(Met->Res[0].Val.V_DOUBLE));  

}
static int CHK_OPT(CF_LibAffCir1d_Fourier_Swaption)(void *Opt, void *Mod)
{
    if ((strcmp(((Option *)Opt)->Name, "PayerSwaption") == 0) || (strcmp(((Option  

        return OK;  

    else  

        return WRONG;  

}
#endif //PremiaCurrentVersion

static int MET(Init)(PricingMethod *Met, Option *Opt)
{

```

```

if (Met->init == 0)
{
    Met->init = 1;
    Met->HelpFilenameHint = "cf_libor_affine_cir1d_swaption_fourier";
}
return OK;
}

PricingMethod MET(CF_LibAffCir1d_Fourier_Swaption) =
{
    "CF_LibAffCir1d_Fourier_Swaption",
    {{ " ", PREMIA_NULLTYPE, {0}, FORBID}},
    CALC(CF_LibAffCir1d_Fourier_Swaption),
    {{ "Price", DOUBLE, {100}, FORBID}, { " ", PREMIA_NULLTYPE, {0}, FORBID}},
    CHK_OPT(CF_LibAffCir1d_Fourier_Swaption),
    CHK_ok,
    MET(Init)
} ;

```