

[Help](#)

```
#include "
href../../../../mod/libor_affine_gould/libor_affine_gould_std/libor_affine_gould_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_LibAffGould_Fourier_CapFloor)(void *Opt, void *Mod)
{
    return NONACTIVE;
}
int CALC(CF_LibAffGould_Fourier_CapFloor)(void *Opt, void *Mod, PricingMethod *M
{
    return AVAILABLE_IN_FULL_PREMIA;
}
#else

static int cf_capfloor_fourier_libaff_gould(int InitYieldCurve_flag, double R_fl
{
    double caplet_Tk1, caplet_Tk2, caplet_price;
    int i, nb_payment = pnl_iround((cap_end - cap_start) / cap_period);
    StructLiborAffine LiborAffine;
    ZCMarketData ZCMarket;

    PnlVect *ModelParams = pnl_vect_create(4);
    LET(ModelParams, 0) = x0;
    LET(ModelParams, 1) = lambda;
    LET(ModelParams, 2) = alpha;
    LET(ModelParams, 3) = beta;

    ZCMarket.filename = curve;
    SetInitYieldCurve(InitYieldCurve_flag, R_flat, &ZCMarket);

    CreateStructLiborAffine(&LiborAffine, &ZCMarket, cap_start, cap_end, cap_perio

    caplet_Tk1 = cap_start;
```

```

caplet_Tk2 = caplet_Tk1 + cap_period;
*cap_price = 0.;

for (i = 0; i < nb_payement; i++)
{
    caplet_price = cf_swaption_fourier_libaff(&LiborAffine, caplet_Tk1, caplet
    *cap_price += caplet_price;

    caplet_Tk1 += cap_period;
    caplet_Tk2 += cap_period;
}

FreeStructLiborAffine(&LiborAffine);

return OK;
}

```

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

```

int CALC(CF_LibAffGould_Fourier_CapFloor)(void *Opt, void *Mod, PricingMethod *M
{
    TYPEOPT *ptOpt = (TYPEOPT *)Opt;
    TYPEMOD *ptMod = (TYPEMOD *)Mod;

    int cap_floor = (((ptOpt->PayOff.Val.V_NUMFUNC_1)->Compute) == &Call);

    return  cf_capfloor_fourier_libaff_gould(ptMod->flat_flag.Val.V_INT,
        MOD(GetYield)(ptMod),
        MOD(GetCurve)(ptMod),
        ptMod->x0.Val.V_DOUBLE,
        ptMod->lambda.Val.V_PDOUBLE,
        ptMod->alpha.Val.V_DOUBLE,
        ptMod->beta.Val.V_PDOUBLE,
        ptOpt->FirstResetDate.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,
        cap_floor,
        &(Met->Res[0].Val.V_DOUBLE));
}

```

```

}
static int CHK_OPT(CF_LibAffGould_Fourier_CapFloor)(void *Opt, void *Mod)
{
    if ((strcmp(((Option *)Opt)->Name, "Cap") == 0) || (strcmp(((Option *)Opt)->Na
        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_gould_capfloor_fourier";
    }
    return OK;
}

```

```

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

```