

[Help](#)

```

#if defined(PremiaCurrentVersion) && PremiaCurrentVersion < (2011+2) //The "#els
#else

#include "pnl/pnl_complex.h"
#include "libor_affine_framework.h"

// Moment generating function of X(Ti) under the for forward measure P(Tk)
dcomplex MomentGF_XTi_PTk(dcomplex v, double Ti, double Tk, StructLiborAffine *L
{
    double x0 = GET(LiborAffine->ModelParams, 0);
    double TN = GET(LiborAffine->TimeDates, (LiborAffine->TimeDates)->size - 1);
    int k = indiceTimeLiborAffine(LiborAffine, Tk);

    dcomplex uk = Complex(GET(LiborAffine->MartingaleParams, k), 0);
    dcomplex phi_i, psi_i, phi_i1, psi_i1, phi_i2, psi_i2, z1, z2, z3, result;

    phi_psi_t_v(TN - Ti, uk, LiborAffine, &phi_i, &psi_i);
    z1 = Cadd(psi_i, v);

    phi_psi_t_v(Ti, psi_i, LiborAffine, &phi_i1, &psi_i1);
    phi_psi_t_v(Ti, z1, LiborAffine, &phi_i2, &psi_i2);

    z2 = Csub(phi_i2, phi_i1);
    z3 = Csub(psi_i2, psi_i1);

    result = Cadd(z2, CRmul(z3, x0));

    return Cexp(result);
}

// Moment generating function of X(Ti) under the for forward measure P(TN)
dcomplex MomentGF_XTi_PTN(dcomplex z, double Ti, StructLiborAffine *LiborAffine)
{
    double x0 = GET(LiborAffine->ModelParams, 0);
    dcomplex phi, psi, result;

    phi_psi_t_v(Ti, z, LiborAffine, &phi, &psi);

    result = Cadd(phi, CRmul(psi, x0));

```

```

    return Cexp(result);
}

// Calibration of martingale parameters to match the initial zero coupon curve.
void CreateStructLiborAffine(StructLiborAffine *LiborAffine, ZCMarketData *ZCMarketData,
                             double T0, double TN, double Period, PnlVect *ModelParams,
                             void (*phi_psi)(PnlVect *ModelParams, double t, double dco),
                             double (*MaxMgfArg)(PnlVect *, double))
{
    double precision = 1e-10, precision_u = 1e-14;
    double Tk, u, u_inf, u_sup;
    double Bond_TN, DisctBond_Tk, Martingale_u;

    int k, N = (TN - T0) / Period;

    LiborAffine->TimeDates = pnl_vect_create(N + 1);
    LiborAffine->MartingaleParams = pnl_vect_create(N + 1);

    for (k = 0; k <= N; k++) LET(LiborAffine->TimeDates, k) = T0 + k * Period;

    LiborAffine->phi_psi = phi_psi;
    LiborAffine->MaxMgfArg = MaxMgfArg;
    LiborAffine->ModelParams = ModelParams;
    LiborAffine->ZCMarket = ZCMarket;

    ///**** Calibration of martingale parameters to match the initial zero coupon curve
    u_inf = 0.;
    u_sup = MaxMgfArg(ModelParams, TN);
    Bond_TN = BondPrice(TN, ZCMarket);

    for (k = 0; k < N; k++)
    {
        Tk = GET(LiborAffine->TimeDates, k);
        DisctBond_Tk = BondPrice(Tk, ZCMarket) / Bond_TN;
        do
        {
            u = 0.5 * (u_inf + u_sup);
            Martingale_u = Creal(MomentGF_XTi_PTN(Complex(u, 0.), TN, LiborAffine));

            if (Martingale_u < DisctBond_Tk) u_inf = u;
        }
    }
}

```

```
        else u_sup = u;
    }
    while (fabs(Martingale_u - DisctBond_Tk) > precision && fabs(u_sup - u_inf) > precision)
    {
        LET(LiborAffine->MartingaleParams, k) = u;

        u_inf = 0.;
        u_sup = u;
    }

    LET(LiborAffine->MartingaleParams, N) = 0.;
    //pnl_vect_print(LiborAffine->MartingaleParams);
}

void FreeStructLiborAffine(StructLiborAffine *LiborAffine)
{
    pnl_vect_free(&(LiborAffine->TimeDates));
    pnl_vect_free(&(LiborAffine->MartingaleParams));
    pnl_vect_free(&(LiborAffine->ModelParams));
    DeleteZCMarketData(LiborAffine->ZCMarket);
}

int indiceTimeLiborAffine(StructLiborAffine *LiborAffine, double s)
{
    double eps = 1e-6;
    int i = 0, N = (LiborAffine->TimeDates)->size - 1;

    while (i <= N && fabs(GET(LiborAffine->TimeDates, i) - s) > eps) i++;

    return i;
}

#endif
```