

[Help](#)

```
#include "
href../../../../mod/cir1d/cir1d_std/cir1d_std_h_src.pdfcir1d_std.h"

static double A, B, h;

/*Zero Coupon Bond*/
static double A_f(double time, double k, double h, double sigma, double theta)
{
    return pow(h * exp(0.5 * (k + h) * (time))) / (h + 0.5 * (k + h) * (exp(h * (time)) - 1.));
}

static double B_f(double time, double k, double h, double sigma, double theta)
{
    return (exp(h * (time)) - 1.) / (h + 0.5 * (k + h) * (exp(h * (time)) - 1.));
}

static double zcb_cir1d(double r0, double k, double t, double sigma, double theta)
{
    h = sqrt(SQR(k) + 2.*SQR(sigma));
    B = B_f(T - t, k, h, sigma, theta);
    A = A_f(T - t, k, h, sigma, theta);

    return A * exp(-B * r0);
}

/*Put Option*/
static int zbp_cir1d(double r, double k, double t, double sigma, double theta, double S)
{
    double K;
    double PtS, PtT, ATS, BTS;
    double p1, p2, p3, k1, k2, k3, psi, phi, rb;

    /*P(t,S)*/
    PtS = zcb_cir1d(r, k, t, sigma, theta, S);
    BTS = B_f(S - T, k, h, sigma, theta);
    ATS = A_f(S - T, k, h, sigma, theta);
```

```

/*P(t,T)*/
PtT = zcb_cir1d(r, k, t, sigma, theta, T);

/*X^2 parameters*/
K = p->Par[0].Val.V_DOUBLE;
rb = log(ATS / K) / BTS;
h = 2.*h;
phi = 2.*h / (SQR(sigma) * (exp(h * (T - t)) - 1.));
psi = (k + h) / SQR(sigma);

p1 = 2.*rb * (phi + psi + BTS);
p2 = 4.*k * theta / SQR(sigma);

p3 = (2.*SQR(phi) * r * exp(h * (T - t))) / (phi + psi + BTS);
k1 = 2.*rb * (phi + psi);
k2 = p2;
k3 = (2.*SQR(phi) * r * exp(h * (T - t))) / (phi + psi);

/*Price of Put by Parity*/
*price = PtS * pnl_cdfchi2n(p1, p2, p3) - K * PtT * pnl_cdfchi2n(k1, k2, k3) -

/*Delta*/
*delta = 0.;

return OK;
}

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

    return zbp_cir1d(ptMod->r.Val.V_PDOUBLE, ptMod->k.Val.V_DOUBLE, ptMod->T.Val.
        ptMod->theta.Val.V_PDOUBLE, ptOpt->BMaturity.Val.V_DATE, ptOp
        &(Met->Res[0].Val.V_DOUBLE), &(Met->Res[1].Val.V_DOUBLE));
}

static int CHK_OPT(CF_ZCPutBondEuro)(void *Opt, void *Mod)
{
    return strcmp(((Option *)Opt)->Name, "ZeroCouponPutBondEuro");
}

```

```

}

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

    return OK;
}

PricingMethod MET(CF_ZCPutBondEuro) =
{
    "CF_Cir1d_ZBPutEuro",
    {{ " ", PREMIA_NULLTYPE, {0}, FORBID}},
    CALC(CF_ZCPutBondEuro),
    {{ "Price", DOUBLE, {100}, FORBID}, {"Delta", DOUBLE, {100}, FORBID} , {" ", PR
    CHK_OPT(CF_ZCPutBondEuro),
    CHK_ok,
    MET(Init)
} ;

```