

Help

```
extern "C" {
#include "jarrowyildirim1d_std.h"
#include "pnl/pnl_cdf.h"
#include "optype.h"
    extern char premia_data_dir[MAX_PATH_LEN];
    extern char *path_sep;
}

#if defined(PremiaCurrentVersion) && PremiaCurrentVersion < (2008+2) //The "#els
#else

static double *tm, *PN, *PR, *ZCSR, *ZCSRT, *tm_zcsr, *PNT;
static int Nvalue, Nvalue1; /*Number of value read for PN*/
static char init[] = "nominal_zcb_prices.dat";
static char init1[] = "zcii_swap_rates.dat";
static FILE *Entrees; /*File variable of the code*/
static FILE *Entrees1;

/*Read Nominal Zero Coupon Bond*/
static int lecture_PN()
{

    int i;
    char ligne[20];
    char *pligne;
    double p, tt;
    char data[MAX_PATH_LEN];

    sprintf(data, "%s%s%s", premia_data_dir, path_sep, init);
    Entrees = fopen(data, "r");

    if (Entrees == NULL)
    {
        printf("Le FICHER %s N'A PU ETRE OUVERT. VERIFIER LE CHEMIN\ n", data);
    }

    i = 0;
    pligne = ligne;
```

```
PN = new double[100];
PNT = new double[100];
tm = new double[100];
PR = new double[100];

while (1)
{
    pligne = fgets(ligne, sizeof(ligne), Entrees);
    if (pligne == NULL) break;
    else
    {
        sscanf(ligne, "%lf t=%lf", &p, &tt);

        PN[i] = p;
        tm[i] = tt;
        i++;
    }
}
Nvalue = i;
fclose(Entrees);

return i;
}

/*Read Zero Coupon Swap Rates*/
static int lecture_ZCSR()
{
    int i;
    char ligne[20];
    char *pligne;
    double p, tt;
    char data[MAX_PATH_LEN];

    sprintf(data, "%s%s%s", premia_data_dir, path_sep, init1);
    Entrees1 = fopen(data, "r");

    ZCSR = new double[100];
    ZCSRT = new double[100];
    tm_zcsr = new double[100];
    if (Entrees1 == NULL)
```

```
{
    printf("Le FICHIER %s N'A PU ETRE OUVERT. VERIFIER LE CHEMIN\ n", data);
}

i = 0;
pligne = ligne;

while (1)
{
    pligne = fgets(ligne, sizeof(ligne), Entrees1);
    if (pligne == NULL) break;
    else
    {
        sscanf(pligne, "%lf t=%lf", &p, &tt);
        ZCSR[i] = p;
        tm_zcsr[i] = tt;

        i++;
    }
}
Nvalue1 = i;
fclose(Entrees1);

return i;
}

static double bond_zcn(double T)
{
    double POT;
    int i = 0;

    if (T > 0)
    {
        while (tm[i] < T && i < Nvalue)
        {
            i = i + 1;
        }

        if (i == 0)
        {
```

```

        POT = 1 * (1 - T / tm[0]) + PN[0] * (T / tm[0]);
    }
    else
    {
        if (i < Nvalue)
        {

            POT = PN[i - 1] * (tm[i] - T) / (tm[i] - tm[i - 1]) + PN[i] * (T -
/*printf("values %d %f %f %f %f\ n",i,PN[i-1],PN[i],tm[i-1],tm[i])
            }
        else
        {
            POT = PN[i - 1] + (T - tm[i - 1]) * (PN[i - 1] - PN[i - 2]) / (tm[
        }
    }
}
else
{
    POT = 1;
}

return POT;
}
static double bond_zcsr(double T)
{
    double POT;
    int i = 0;

    if (T > 0)
    {
        while (tm_zcsr[i] < T && i < Nvalue1)
        {
            i = i + 1;
        }

        if (i == 0)
        {
            POT = 1 * (1 - T / tm_zcsr[0]) + ZCSR[0] * (T / tm_zcsr[0]);
        }
    else
    {

```

```

        if (i < Nvalue)
        {
            POT = ZCSR[i - 1] * (tm_zcsr[i] - T) / (tm_zcsr[i] - tm_zcsr[i - 1])
        }
        else
        {
            POT = ZCSR[i - 1] + (T - tm_zcsr[i - 1]) * (ZCSR[i - 1] - ZCSR[i - 2]) / (tm_zcsr[i] - tm_zcsr[i - 1])
        }
    }
}
else
{
    POT = 0;
}
return POT;
}

```

```

/*Compute ZeroCoupon Bond Price in Creal Economy*/
static void CalculatePR(int tenor_order, double tenor, double swap_mat)
{
    int i, j;

    i = lecture_PN();
    j = lecture_ZCSR();
    i = MIN(i, j);
    if (swap_mat > tm[i - 1])
    {
        printf("\nError : time bigger than the last time value entered in market\n");
        exit(EXIT_FAILURE);
    }
    else
    {
        PNT[0] = 1.;
        for (int j = 1; j < tenor_order + 1; j++)
        {
            PNT[j] = bond_zcn((double)j * tenor);
            ZCSRT[j] = bond_zcsr((double)j * tenor);
        }

        PR[0] = 1.0;
    }
}

```

```

        for (int j = 1; j < tenor_order + 1; j++)
        {
            PR[j] = PNT[j] * pow((1.0 + ZCSRT[j]), (double)j * tenor);
            /*printf("%f\ n",PR[j]);*/
        }
    }
}

/*Compute Function Beta in Page 91 of Moreni's thesis (Function B in page 16 of sl
static double Beta(double a, double t1, double t2)
{
    double beta = 0.0;
    if ((t2 - t1) == 0.0)
    {
        beta = 1.0;
    }
    else
    {
        beta = (1.0 - exp(-a * (t2 - t1))) / a;
    }
    return beta;
}

/*compute function gamma in page 92 of Moreni's thesis (function C in page 17 of
static double ParameterGamma(double t, double tenor, int caplet_number, double

{
    double result = (sigmar * Beta(ar, (caplet_number - 1) * tenor, caplet_number
    return result;
}

static double Variance(double t, int caplet_number, double sigman, double sigmar
{
    double VarianceN = sigman * sigman / (2.0 * pow(an, 3)) * pow((1.0 - exp(-an *
    double VarianceR = sigmar * sigmar / (2.0 * pow(ar, 3)) * pow((1.0 - exp(-ar *
    double CorelateNR = 2.0 * rhonr * sigman * sigmar / (an * ar * (an + ar)) * (1
    double VarianceCPI = sigma_cpi * sigma_cpi * (tm[caplet_number] - tm[caplet_nu
    double VarianceNN = sigman * sigman / an / an * (tm[caplet_number] - tm[caplet
    double VarianceRR = sigmar * sigmar / ar / ar * (tm[caplet_number] - tm[caplet
    double CorelateNNRR = 2.0 * rhonr * sigman * sigmar / (an * ar) * (tm[caplet_n
    double CorelateNCPI = 2.0 * rhoncpi * sigman * sigma_cpi / an * (tm[caplet_num
    double CorelateRCPI = 2.0 * rhorcpi * sigmar * sigma_cpi / ar * (tm[caplet_num

```

```

    return double(VarianceN + VarianceR - CorelateNR + VarianceCPI + VarianceNN +
}

static double Mean(double t, double tenor, int caplet_number, double sigman, dou
{
    double result = PN[caplet_number - 1] / PN[caplet_number] * PR[caplet_number]
    return result;
}

static double Positiveb(double variance, double mean, double strike)
{
    return double((log(mean / (strike + 1.0)) + pow(variance, 2) / 2.0) / variance
}

static double Negativeb(double variance, double mean, double strike)
{
    return double((log(mean / (strike + 1.0)) - variance * variance / 2.0) / varia
}

static int cf_iicaplet1d(NumFunc_1 *p, double t, double caplet_maturity, double
{
    int caplet_number = (int)((caplet_maturity - t) / tenor);
    int omega = 1;

    /*Compute ZeroCoupon Bond Price in Creal Economy*/
    CalculatePR(caplet_number, tenor, caplet_maturity);

    double variance, mean, bposi, bnega, dfposi, dfnega; //temporary variables
    variance = Variance(t, caplet_number, sigman, sigmar, sigma_cpi, an, ar, rhon
    mean = Mean(t, tenor, caplet_number, sigman, sigmar, sigma_cpi, an, ar, rhon
    bposi = Positiveb(variance, mean, strike);
    bnega = Negativeb(variance, mean, strike);
    dfposi = cdf_nor(omega * bposi);
    dfnega = cdf_nor(omega * bnega);

    /*Price*/
    *price = omega * PN[caplet_number] * (mean * dfposi - (strike + 1) * dfnega);

    delete [] tm;
    delete [] PN;
    delete [] PNT;

```

```

    delete [] PR;
    delete [] ZCSR;
    delete [] ZCSRT;
    delete [] tm_zcsr;

    return OK;
}
#endif //PremiaCurrentVersion

extern "C" {
#if defined(PremiaCurrentVersion) && PremiaCurrentVersion < (2008+2) //The "#els
    static int CHK_OPT(CF_YI_IICAPLET)(void *Opt, void *Mod)
    {
        return NONACTIVE;
    }
    int CALC(CF_YI_IICAPLET)(void *Opt, void *Mod, PricingMethod *Met)
    {
        return AVAILABLE_IN_FULL_PREMIA;
    }
#else
    int CALC(CF_YI_IICAPLET)(void *Opt, void *Mod, PricingMethod *Met)
    {
        TYPEOPT *ptOpt = (TYPEOPT *)Opt;
        TYPEMOD *ptMod = (TYPEMOD *)Mod;

        return cf_iicaplet1d(ptOpt->PayOff.Val.V_NUMFUNC_1, ptMod->T.Val.V_DATE, ptO
    }

    static int CHK_OPT(CF_YI_IICAPLET)(void *Opt, void *Mod)
    {

        if ((strcmp(((Option *)Opt)->Name, "InflationIndexedCaplet") == 0))
            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_JarrowYildirim1d_iiCaplet";

}
return OK;
}

PricingMethod MET(CF_YI_IICAPLET) =
{
    "CF_JarrowYildirim1d_iiCap",
    {{" ", PREMIA_NULLTYPE, {0}, FORBID}},
    CALC(CF_YI_IICAPLET),
    {"Price", DOUBLE, {100}, FORBID} , {" ", PREMIA_NULLTYPE, {0}, FORBID}},
    CHK_OPT(CF_YI_IICAPLET),
    CHK_ok,
    MET(Init)
} ;
}
```