

[Help](#)

```
#if defined(PremiaCurrentVersion) && PremiaCurrentVersion < (2007+2) //The "#els
#else

/// \ file cdsCirpp.h
/// \ brief cds_spread_CIRPPMC and cds_spread_GaussMap functions
/// \ author M. Ciuca (MathFi, ENPC)
/// \ note (C) Copyright Premia 8 - 2006, under Premia 8 Software license
//
// Use, modification and distribution are subject to the
// Premia 8 Software license

#ifndef _CDS_CIRPP_H
#define _CDS_CIRPP_H

#include <stdexcept>
#include <iostream>
#include <fstream>
#include <iomanip>
#include <string>
#include <vector>
#include <math.h>

double cds_spread_CIRPPMC_MKT( // Computes the value of the spread which correspo
    double maturity, // maturity of the CDS (in years)
    int period, // payment period, in months
    double recovery, // expected recovery rate
    std::vector<double> &RatesMat, // Maturities of zero-coupons for calibration
    std::vector<double> &Rates, // rates of risk-free zero-coupons for calibration
    std::vector<double> &intensityMat, // Maturities of CDS used for calibration
    std::vector<double> &intensityRates, // intensity of the name underlying the
    double &DefaultLeg, // DefaultLeg price (return parameter)
    double &PaymentLeg // PaymentLeg price (return parameter)
);

// Very simple calibration of default intensity.
```

```

// Characteristics:
// 1. Interest rates are flat
// 2. The calibrated intensity is piecewise linear
// 3. The input spreads curve cannot have more than 5 spreads
void DefaultIntensityCalibration( // Computes the implied deterministic default
    double recovery, // expected recovery rate
    int period, // payment period, in months
    std::vector<double> &spreadsMat, // Maturities of CDS used for calibration
    std::vector<double> &spreads, // spreads of CDS used for calibration
    double r, // instantaneous interest rate (flat interest rates)
    std::vector<double> &intensityMat, // time grid points from the calibrated intensity
    std::vector<double> &intensityRates // intensity of the name underlying the CDS
);

/*
double cds_spread_CIRPPMC( // Computes the value of the spread which corresponds to
    double maturity, // maturity of the CDS (in years)
    int period, // payment period, in months
    double recovery, // expected recovery rate
    double precision, // time step for CIR processes path simulation scheme
    int Nsim, // number of Monte Carlo simulations
    double mrRate, // mean reversion coefficient in the interest rate model
    double mrIntensity, // mean reversion coefficient in the intensity model
    double sigmaRate, // volatility coefficient in the interest rate model
    double sigmaIntensity, // volatility coefficient in the intensity model
    double thetaRate, // long-run mean in the interest rate model
    double thetaIntensity, // long-run mean in the intensity model
    double x0_r, // Starting value of the short rate process
    double x0, // Starting value of the intensity process
    double correlation, // correlation between rate and intensity
    std::vector<double> & RatesMat, // Maturities of zero-coupons for calibration
    std::vector<double> & Rates, // rates of risk-free zero-coupons for calibration
    std::vector<double> & intensityMat, // Maturities of CDS used for calibration
    std::vector<double> & intensityRates, // intensity of the name underlying the CDS
    double& DefaultLeg, // DefaultLeg price (return parameter)
    double& PaymentLeg, // PaymentLeg price (return parameter)
    double& std_dev_DefaultLeg, // DefaultLeg standard deviation (return parameter)
    double& std_dev_PaymentLeg, // PaymentLeg standard deviation (return parameter)
    double barrier = 1.0 // Barrier for the intensity process
);
*/

```

```

double cds_spread_CIRPPMC_CV( // Computes the value of the spread which corresponds to
    double maturity, // maturity of the CDS (in years)
    int period, // payment period, in months
    double recovery, // expected recovery rate
    double precision, // time step for CIR processes path simulation scheme
    int Nsim, // number of Monte Carlo simulations
    double mrRate, // mean reversion coefficient in the interest rate model
    double mrIntensity, // mean reversion coefficient in the intensity model
    double sigmaRate, // volatility coefficient in the interest rate model
    double sigmaIntensity, // volatility coefficient in the intensity model
    double thetaRate, // long-run mean in the interest rate model
    double thetaIntensity, // long-run mean in the intensity model
    double x0_r, // Starting value of the short rate process
    double x0, // Starting value of the intensity process
    double correlation, // correlation between rate and intensity
    std::vector<double> &RatesMat, // Maturities of zero-coupons for calibration
    std::vector<double> &Rates, // rates of risk-free zero-coupons for calibration
    std::vector<double> &intensityMat, // Maturities of CDS used for calibration
    std::vector<double> &intensityRates, // intensity of the name underlying the
    double &DefaultLeg, // DefaultLeg price (return parameter)
    double &PaymentLeg, // PaymentLeg price (return parameter)
    double &std_dev_DefaultLeg, // DefaultLeg standard deviation (return parameter)
    double &std_dev_PaymentLeg, // PaymentLeg standard deviation (return parameter)
    double barrier, // Barrier for the intensity process
    int generator
);

```

```

double cds_spread_GaussMap( // Computes the value of the spread which corresponds to
    double maturity, // maturity of the CDS
    int period, // payment period, in months
    double recovery, // expected recovery rate
    double mrRate, // mean reversion coefficient in the interest rate model
    double mrIntensity, // mean reversion coefficient in the intensity model
    double sigmaRate, // volatility coefficient in the interest rate model
    double sigmaIntensity, // volatility coefficient in the intensity model
    double thetaRate, // long-run mean in the interest rate model

```

```

double thetaIntensity, // long-run mean in the intensity model
double x0_r, // Starting value of the short rate process
double x0, // Starting value of the intensity process
double correlation, // correlation between rate and intensity
std::vector<double> &RatesMat, // Maturities of zero-coupons for calibration
std::vector<double> &Rates, // rates of risk-free zero-coupons for calibration
std::vector<double> &intensityMat, // Maturities of CDS used for calibration
std::vector<double> &intensityRates, // intensity of the name underlying the
double &DefaultLeg, // DefaultLeg price (return parameter)
double &PaymentLeg // PaymentLeg price (return parameter)
);

/*
double cds_spread_GaussMap( // Computes the value of the spread which correspond
double maturity, // maturity of the CDS
int period, // payment period, in months
double recovery, // expected recovery rate
double mrRate, // mean reversion coefficient in the interest rate model
double mrIntensity, // mean reversion coefficient in the intensity model
double sigmaRate, // volatility coefficient in the interest rate model
double sigmaIntensity, // volatility coefficient in the intensity model
double thetaRate, // long-run mean in the interest rate model
double thetaIntensity, // long-run mean in the intensity model
double y0, // Starting value of the intensity process
double correlation, // correlation between rate and intensity
std::vector<double> & RatesMat, // Maturities of zero-coupons for calibration
std::vector<double> & Rates, // rates of risk-free zero-coupons for calibration
std::vector<double> & spreadMat, // Maturities of CDS used for calibration
std::vector<double> & spreadRates); // spreads of CDS for calibration
*/
#endif

#endif //PremiaCurrentVersion

```