```
   Help
#if defined(PremiaCurrentVersion) && PremiaCurrentVersion <
     (2007+2) //The "#else" part of the code will be freely av
    ailable after the (year of creation of this file + 2)
#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 Sof
    tware 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 th
    e spread which corresponds to zero price CDS
  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 th
```

```
    e name underlying the CDS; (spreads of CDS for calibration)
  double &DefaultLeg, // DefaultLeg price (return paramete
    r)
  double &PaymentLeg // PaymentLeg price (return parameter)
);


// Very simple calibration of default intensity.
// Characteristics:
// 1. Interest rates are flat
// 2. Tha calibrated intensity is piecewise linear
// 3. The input spreads curve cannot have more then 5 sprea
    ds
void DefaultIntensityCalibration( // Computes the implied
    deterministic default intensity from a CDS spreads curve
  double recovery, // expected recovery rate
  int period, // payment period, in months
  std::vector<double> &spreadsMat,  // Maturities of CDS us
    ed for calibration
  std::vector<double> &spreads,  // spreads of CDS used fo
    r calibration
  double r, // instantaneous interest rate (flat interst ra
    tes)
  std::vector<double> &intensityMat,  // time greed points
    from the calibrated intensity (return parameter)
  std::vector<double> &intensityRates  // intensity of the
    name underlying the CDS curve (return parameter)
);

/*
double cds_spread_CIRPPMC( // Computes the value of the spr
    ead which corresponds to zero price CDS
              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 simu
    lations
              double mrRate, // mean reversion coeffi
```

```
    cient in the interest rate model
              double mrIntensity, // mean reversion
    coefficient in the intensity model
              double sigmaRate, // volatility coeffi
    cient 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 mea
    n 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 bet
    ween rate and intensity
              std::vector<double> & RatesMat, // Matu
    rities 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; (spreads of     CDS for calibrat
              double& DefaultLeg, // DefaultLeg
    price (return parameter)
              double& PaymentLeg, // PaymentLeg
    price (return parameter)
              double& std_dev_DefaultLeg, // Default
    Leg standard deviation (return parameter)
              double& std_dev_PaymentLeg, // Payment
    Leg standard deviation (return parameter)
              double barrier = 1.0 // Barrier for th
    e intensity process
              );
              */



double cds_spread_CIRPPMC_CV( // Computes the value of the
```

```
    spread which corresponds to zero price CDS
  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 si
    mulation 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 intere
    st 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 th
    e name underlying the CDS; (spreads of CDS for calibration)
  double &DefaultLeg, // DefaultLeg price (return paramete
    r)
  double &PaymentLeg, // PaymentLeg price (return paramete
    r)
  double &std_dev_DefaultLeg, // DefaultLeg standard devi
    ation (return parameter)
  double &std_dev_PaymentLeg, // PaymentLeg standard devi
    ation (return parameter)
  double barrier,// Barrier for the intensity process
  int generator
);
```

```
double cds_spread_GaussMap( // Computes the value of the
    spread which corresponds to zero price CDS
  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 intere
    st 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 th
    e name underlying the CDS; (spreads of CDS for calibration)
  double &DefaultLeg, // DefaultLeg price (return paramete
    r)
  double &PaymentLeg // PaymentLeg price (return parameter)
);



/*
double cds_spread_GaussMap( // Computes the value of the
    spread which corresponds to zero price CDS
          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 coeffi
    cient in the intensity model
          double sigmaRate, // volatility coefficient
    in the interest rate model
          double sigmaIntensity, // volatility coeffi
    cient in the intensity model
          double thetaRate, // long-run mean in the
    interest rate model
          double thetaIntensity, // long-run mean in th
    e intensity model
          double y0, // Starting value of the
    intensity process
          double correlation, // correlation between ra
    te and intensity
          std::vector<double> & RatesMat, // Maturitie
    s of zero-coupons for calibration
          std::vector<double> & Rates, // rates of ris
    k-free zero-coupons for calibration
          std::vector<double> & spreadMat, // Maturitie
    s of CDS used for calibration
          std::vector<double> & spreadRates); // sprea
    ds of CDS for calibration
*/
#endif

#endif //PremiaCurrentVersion
```

# References