NAME

BinomialLossModel< LLM >

SYNOPSIS

`#include <ql/experimental/credit/binomiallossmodel.hpp>`

Inherits **DefaultLossModel**.

Public Types

typedef LLM::copulaType **copulaType**

Public Member Functions

BinomialLossModel (const boost::shared_ptr< LLM > &copula)

Protected Member Functions

Disposable< std::vector< **Real** > > **expectedDistribution** (const **Date** &date) const

Disposable< std::vector< **Real** > > **lossPoints** (const **Date** &) const
attainable loss points this model provides

Disposable< std::map< **Real**, **Probability** > > **lossDistribution** (const **Date** &d) const
Returns the cumulative full loss distribution.

Real percentile (const **Date** &d, **Real** percentile) const
Loss level for this percentile.

Real expectedShortfall (const **Date** &d, **Real percentile**) const
Expected shortfall given a default loss percentile.

Real expectedTrancheLoss (const **Date** &d) const

Real averageLoss (const **Date** &, const std::vector< **Real** > &reminingNots, const std::vector< **Real** > &) const
Average loss per credit.

Real condTrancheLoss (const **Date** &, const std::vector< **Real** > &lossVals, const std::vector< **Real** > &bsktNots, const std::vector< **Probability** > &uncondDefProbs, const std::vector< **Real** > &) const

Disposable< std::vector< **Real** > > **expConditionalLgd** (const **Date** &d, const std::vector< **Real** > &mktFactors) const

Disposable< std::vector< **Real** > > **lossProbability** (const **Date** &date, const std::vector< **Real** > &bsktNots, const std::vector< **Real** > &uncondDefProbInv, const std::vector< **Real** > &mktFactor) const
Loss probability density conditional on the market factor value.

Protected Attributes

const boost::shared_ptr< LLM > **copula_**

Real attachAmount_

Real detachAmount_

Detailed Description

template<class LLM>

class QuantLib::BinomialLossModel< LLM >" Binomial Defaultable **Basket** Loss Model

Models the portfolio loss distribution by approximatting it to an adjusted binomial. Fits the two moments of the loss distribution through an adapted binomial approximation. This simple model allows for portfolio inhomogeneity with no excesive cost over the LHP.

See:

Approximating Independent Loss Distributions with an Adjusted Binomial Distribution , Dominic O'Kane, 2007 EDHEC RISK AND ASSET MANAGEMENT RESEARCH CENTRE

Modelling single name and multi-name credit derivatives Chapter 18.5.2, Dominic O'Kane, Wiley Finance, 2008

The version presented here is adaptated to the multifactorial case by computing a conditional binomial approximation; notice that the Binomial is stable. This way the model can be used also in

risk management models rather than only in pricing. The copula is also left undefined/arbitrary.

LLM: Loss Latent Model template parameter able to model default and loss.

The model is allowed and arbitrary copula, although initially designed for a Gaussian setup. If these exotic versions were not allowed the template parameter can then be dropped but the use of random recoveries should be added in some other way.

Member Function Documentation

Disposable<std::vector<Real> > expectedDistribution (const Date & date) const `[protected]`

Returns the probability of the default loss values given by the method lossPoints.

Author

Generated automatically by Doxygen for QuantLib from the source code.