A COMPARISON OF THE METHODS USED TO DETERMINE THE PORTFOLIO CREDIT LOSS DISTRIBUTION AND THE PRICING OF SYNTHETIC CDO TRANCHES

Renay Singh

A dissertation submitted to the Faculty of Science, University of the Witwatersrand, in fulfilment of the requirements for the degree of Master of Science in the Department of Statistics and Actuarial Science

Johannesburg, 2010

Declaration

I declare that this dissertation is the result of my own work. I would like to acknowledge my supervisor, Professor R.J. Thomson who was generous with his ideas and whose attention to detail made this a genuine learning experience for me. The dissertation is being submitted for the Degree of Master of Science in the University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination in any other University. I was employed by Nedbank Capital during the period during which the dissertation was written. The funding by the National Research Foundation is acknowledged.

____

____________________________day of ________________________2010

ABSTRACT

This work aims to provide an introduction to the methodologies used for determining the loss distribution of a heterogeneous portfolio of credit default swaps. For all the methods considered, the theory and the algorithms are presented and their computational efficiency and accuracy investigated. The loss distribution is then used to value synthetic CDO tranches. The multi-step and the default-time approach are the primary methods considered. For the multi-step approach, three approaches in the literature to the computationally demanding task of obtaining the default thresholds are compared. A synthetic CDO tranche was then evaluated and it was found that the choice of method used to determine the default thresholds is significant. The default-time approach was found to be computationally more efficient than the multi-step approach though with significant differences in the tail region of the loss distribution. Both these approaches rely on Monte Carlo simulation, which is computationally inefficient. Semi-analytic approximations to the default-time approach are considered. These are the numerical inversion of the characteristic function, exact recursion and the compound Poisson approximation. A unique presentation that aids in the understanding and implementation of the numerical inversion of the characteristic function is given. The approximation techniques though computationally more efficient than Monte Carlo, are not as accurate.

In memory of my parents

Lalitha Devi Singh

Woodhull Singh

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This research is concerned with the computational techniques used to obtain the probability distribution of losses that are incurred when corporate bonds, bank loans or any other asset default within a portfolio. These assets (from the investors' perspective) are risky in that their return is uncertain. Risk-free assets in this context are bonds issued by a government in its own currency. The loss distribution is essential for the valuation of synthetic collateralised debt obligations (CDOs).

The innovations for the pricing of credit risk have primarily emanated from financial economics. Financial economics is the interaction of three distinct branches of financial risk quantification: mathematical finance, asset-pricing models and corporate finance (Whelan et al, 2002). Whelan et al. (2002) also highlights the importance of financial economics to the actuarial profession.

This introductory chapter is organised as follows. Section 1.1 introduces the notion of credit risk and default risk in particular. In section 1.2 the basic building block of the credit derivatives industry, the credit default swap, is introduced and in section 1.3 a general description of a CDO is given. An important distinguishing characteristic of a CDO is the effect of default correlation. By means of a simple example, section 1.4 demonstrates the importance of default correlation. Section 1.5 provides a rigorous description of the fundamental pricing problem of this research. As will be elucidated in subsequent chapters, actuarial science has made a contribution to the understanding of credit risk in general. Section 1.6 provides such an example where computational efficiency and accuracy is discussed. The chapter concludes with an overview of the subsequent chapters of this dissertation.

1.1 Credit Risk

Duffie & Singleton (2003) provide a definition of credit risk:

> Credit risk is the risk of default or of reductions in market
> value caused by changes in the credit quality of issuers or
> counterparties.

This is a broad definition of credit risk. This research deals with the risk of default. The other type of credit risk the definition alludes to could be interpreted as changes in the credit spread of the company's debt. The credit spread is the excess in interest paid by the corporate over that paid on equivalent risk-free debt.

1.2 Credit default swaps

Credit derivatives are a means of transferring credit risk between two parties by means of bilateral agreements. Arvanitis & Gregory (2001) define a 'credit default swap' (CDS) as a bilateral contract in terms of which an entity buys default protection or insurance with respect to a default event from another, on a predetermined notional amount. The protection buyer will make periodic payments. In return the protection seller will make a payment to compensate the buyer if the credit event does occur. The contract has a given maturity, but will terminate early if default occurs. In the event of default, the contingent amount paid to the buyer is adjusted by a recovery rate; this amount is called the 'loss given default'. This is the amount that the buyer is able to recoup. Credit events other than default are defined by the International Swap and Derivatives Association (Schönbucher, 2003).

1.3 Risk-neutral vs historical default probabilities

The choice of risk measure to use when valuing credit-risk-related instruments such as CDSs depends on the objective (Bluhm et al, 2003). For economic capital or risk-management valuations historical probabilities is the appropriate probability measure. If it is pricing or hedging credit-related securities, then the model is constructed under the risk-neutral probability measure. Under the risk-neutral measure all individuals are indifferent to risk. They require no compensation for risk and the expected return on all securities is the risk-free rate. For credit derivative instruments risk neutrality is achieved by calibrating the default probabilities of individual credits with market-implied default probabilities obtained from bond or CDS spreads. The actual procedure is given in chapter 3.

1.4 Collateralised debt obligations

Bluhm & Wagner (2004) provide a description. A CDO is a financial structured product with two main components, namely an asset leg and a liability leg. The first step in the creation of a CDO is to establish a legal entity, called a special purpose vehicle (SPV), which could be a company, trust or partnership, and which is termed the 'issuer'. This entity is designed to be unaffected by the survival or otherwise of the originator of the assets and is thus termed 'standalone' or 'bankruptcy remote'. In a cash or funded CDO the SPV buys assets from the originator and funds itself by the issuance of securities, which are the SPV's liabilities. The asset leg always consists of a pool of credit-risky instruments, e.g. bonds, loans or CDSs. The securities issued to investors are divided into tranches or blocks of securities that have different risk levels. The investor who buys the riskiest security issued by the SPV will earn the highest running yield.

The two components of a CDO are connected by means of a legal document describing the link between the performance of the asset leg and the performance of tranched securities at the liability leg of the CDO. The performance of a CDO is best illustrated with a cash-flow transaction. The available proceeds from the underlying assets are distributed at the liability leg according to rules defined in the structural documentation of the deal. If the asset pool underlying the CDO performs, i.e. no losses occur, then investors at the liability leg will receive the full promised coupon written on the tranche and a full repayment of their invested capital. But if defaults occur at the asset leg resulting in a loss, then typically some investors will lose money, either in the form of a reduced repayment of invested capital or of a reduced interest payment on their investment. The overall net return to tranche investors attains its maximum if the asset pool performs at its best, but decreases accordingly, if the credit risk inherent in the asset pool leads to defaults and losses. Further descriptive material on cash and synthetic CDOs may be found in Goodman & Fabozzi (2002). Synthetic CDO trading has increased since the inception of the iTraxx and iBoxx standardised tranche indices produced by the

International Index Company (Andersen & Sidenius, unpublished). These indices are an amalgamation of underlying CDSs in various regions and industrial sectors.

1.5 The importance of default correlation

The example of Andersen & Sidenius (unpublished), provides a simple explanation on why default matters. Consider a pool of assets with 100 companies, each of which has a probability of default of 2% over the next year. In the first special case, all the companies are completely independent, implying a default correlation of zero. In this case the probability that a CDO will experience some credit loss over the next year is given by the binomial distribution and is

$$1-\left(1-0.02\right)^{100} = 86.7\%.$$

The probability of a loss in the case of perfect correlation is 2%. Thus the probability that an equity (or synonymously 'junior' or 'first-loss') tranche investor suffering a loss is lower in the case of perfect correlation. In general, the lower the default correlation, the greater the cost to insure the equity tranche against default.

1.6 Pricing CDOs

The following description of synthetic CDOs is based on De Prisco et al. (2005). A synthetic CDO is a CDO whose underlying assets are a portfolio of CDSs. In this instance the issuer of the CDO is the credit protection seller. In return for selling the credit protection the issuer receives periodic premiums. To offset the issuer's risk from these CDSs, part of the total premiums received is allocated to tranches of the CDO. There is a priority scheme for the tranches to absorb the portfolio losses, up to a fixed, maximum amount for each tranche. Losses are based on the recovery-adjusted CDS notional values or the loss given default amounts. The buyer of one or more of these tranches sells partial protection to the issuer, by agreeing to absorb up to the set amount of the portfolio's losses, in exchange for periodic premium payments.

Let $\mathrm{t}_0 = 0, t_1, \ldots, t_n = T$ denote the premium dates, T denoting the maturity of the CDO, and let $B(t)$ be the risk-free discounting factor at time t. Let N_i and R_i denote the notional amount of credit *i* and its recovery rate, respectively. The recovery rate is the amount that is recovered in the event of default. The defaulted asset does not necessarily have zero value. It is assumed that the recovery rates are constant. If the *i*th credit is in default, then the amount $(1-R_i)N_i$, is called the 'loss given default'.

For the purposes of this research, a portfolio of CDSs with identical losses given default, identical default correlations and identical risk-neutral default probabilities is termed 'completely homogeneous'. If only losses given default are identical, then the portfolio is termed 'homogeneous'. In all other cases, the portfolio is termed 'heterogeneous'. The minimum value of each tranche is called the 'attachment' point and the maximum value is called the 'detachment' point. The tranche size is equal to the detachment point less the attachment point. The CDO that is considered in this dissertation consists of three tranches, into which the portfolio's losses are absorbed in accordance with the size and seniority of the tranches. Let N_E, N_M and N_S denote the increasing detachment points for a three-tranche CDO. The intervals $[0, N_E)$, $[N_E, N_M)$and $[N_M, N_S]$ are called the equity (or first loss), mezzanine and senior tranche respectively.

The portfolio cumulative loss at time t is denoted by $L(t)$. For the purpose of this illustration, a single tranche of size S, above the detachment point N_L will be considered. The total loss for this tranche is given by $L^*(t) = \min\left(\max\left(L(t) - N_L, 0\right), S\right)$. Let $\mathbb{E}$ denote the risk-neutral expectation. At inception, the sum of the expected value of the discounted premiums or fair tranche spread for maturity T, s_T, is equal to the sum of the expected value of the discounted losses. This is expressed as

$$s_T \sum_{i=1}^{n} (t_i - t_{i-1})\left(S - \mathbb{E}\left[L_i^*\right]\right) B(t_i) = \sum_{i=1}^{n} \left(\mathbb{E}\left[L_i^*\right] - \mathbb{E}\left[L_{i-1}^*\right]\right) B(t_i)$$

where $L_i^* = L^*(t_i)$ and $L_0^* = 0$. Solving for s_T yields the fair tranche spread:

$$s_T = \frac{\sum_{i=1}^{n}\left(\mathbb{E}\left[L_i^*\right]-\mathbb{E}\left[L_{i-1}^*\right]\right)B(t_i)}{\sum_{i=1}^{n}(t_i-t_{i-1})\left(S-\mathbb{E}\left[L_i^*\right]\right)B(t_i)}. \tag{1.1}$$

The determination of the mean absorbed tranche $\mathbb{E}\left[L_i^*\right]$ and the tranche spread as per equation (1.1) is the focus of this research.

In Figure 1.1 a pictorial description is given for a typical credit loss distribution of the equity, mezzanine and senior (or last loss) tranches. Attention is drawn to the most risky tranche (first loss) and the least risky tranche (the last loss). The equity-tranche investor would expect to receive a higher coupon payment than that received by the senior-tranche investor.

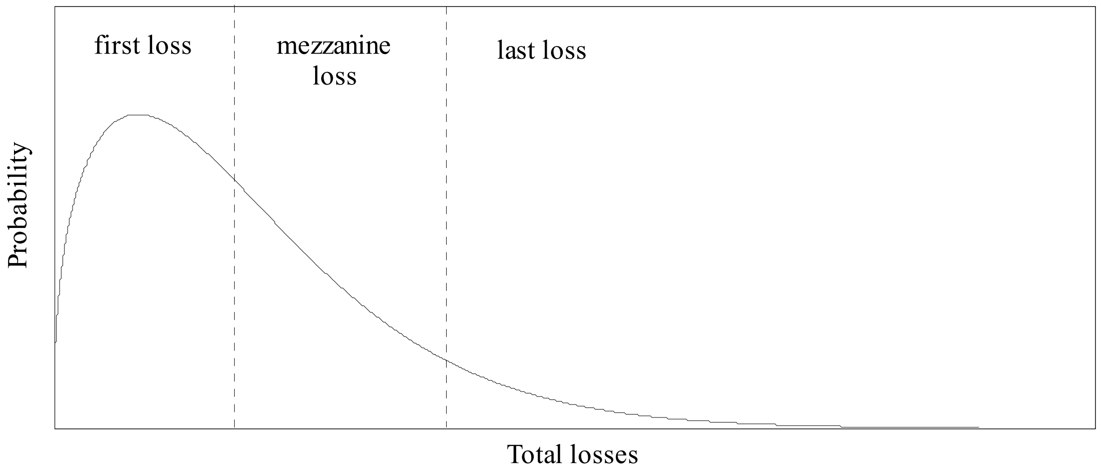

Figure 1.1 The loss distribution and the respective tranches

1.7 Computational efficiency and accuracy

The main objective of this research is to evaluate the fair tranche spreads of heterogeneous CDO portfolios, using different techniques. The results of these different methods are then compared with each other with respect to their accuracy and computational efficiency. The importance of computational efficiency was stressed by Boyle, Broadie & Glasserman (1997):

> In practice, the evaluation of price *sensitivities* is often as important as the evaluation of the prices themselves. Indeed, whereas prices for some securities can be observed in the market, their sensitivities to parameter changes typically cannot and must therefore be computed. Since price sensitivities are important measures of risk, the growing emphasis on risk management systems suggests a greater need for their efficient computation.

In most instances there is a trade-off between accuracy and computational efficiency. From a trading perspective both objectives are important and given the movement of interest rates and credit spreads the synthetic CDO tranches need to be evaluated speedily and accurately. Thus the impetus of researchers has been to determine fast and accurate methods.

The use of Monte Carlo simulation to verify the accuracy of new valuation techniques or financial products has its roots in actuarial science. The pioneer of the technique in finance was Phelim Boyle. In the following extract of an interview the use of Monte Carlo to test the accuracy of another technique and its origins are covered.

> 'The way I picked it up was sitting for the actuarial exams – not the only way to learn it, and maybe not the best but it was there'. When Boyle was sitting for his actuarial exams in the late sixties Monte Carlo wasn't actually 'part of the toolkit'. The British uber-actuary Sidney Benjamin was a one-man pressure group for the adoption of the technique. At the time Boyle was at Irish Life, Monte Carlo was far from the industry practice, but it was in part in the exams 'It was quite enlightened of them,' muses Boyle 'because usually with these professional exams they only teach you stuff that's needed today – it was fortuitous that they included it in the exams.'
>
> Eduardo Schwartz needed his PDE (partial differential equation) checked – Monte Carlo was, to Boyle – if to no one else – an extremely viable tool 'The idea's so simple, you've seen it you remember it – you don't need a big sophisticated course to learn it. Eduardo's PDE took weeks to write, he didn't need Monte Carlo to price the option. But Monte Carlo was a quick and dirty technique to check the results of the PDE – because it was so flexible.'

(Tudball, unpublished).

1.8 Outline of this research

This dissertation is organised as follows. A survey of the literature on credit risk, particularly the valuation of CDOs, is reviewed in chapter 2. Also in chapter 2, the contribution of this research to the literature is stated. An essential ingredient necessary for the valuation of CDOs is the default probabilities of the individual corporate names that comprise the CDO portfolio. A method to extract the default probability from market quotes of CDSs is described and implemented in chapter 3. Chapter 4 is concerned with one of the first approaches to CDO valuation that appeared in the literature. This approach has been found to be computationally too demanding for trading purposes. The default-time approach is described in Chapter 5. It is demonstrated that this approach is a computationally more efficient simulation technique than the multi-step approach. Chapter 6 derives models based on a semi-analytic formulation for the default-time approach. These models are evaluated with techniques that should be familiar to actuaries. Also in this chapter, a unique presentation is given for the numerical inversion of the characteristic function method. This is used to determine the portfolio credit loss distribution. The presentation provides a rigorous discussion of the theory and the practical implementation of the method from a statistical perspective. This was an omission in the literature. The method presented is general enough to be used for known and unknown discrete distributions. Terms used for the various approaches to CDO valuation are defined in the chapters in which they are discussed.

Chapter 2

Literature review

The first synthetic collateralised debt obligation was introduced in 1997 (Smithson, 2004). The rapid growth of the credit derivatives industry (British Bankers' Association, unpublished) and particularly CDOs has led to increased research into efficient methods for the pricing and risk management of synthetic CDO tranches. The resultant literature of models is reviewed in this chapter primarily on the criteria of accuracy and computational efficiency. The need for accuracy and computationally efficiency was captured by De Prisco et al. (2005):

> In principle, a synthetic collateralised debt obligation (CDO) can be valued by Monte Carlo methods. This is perhaps the most common approach used by practitioners today. The problem is that Monte Carlo methods usually take a significant amount of time to achieve the required accuracy. Hence, while such methods may lend themselves to pricing and structuring, they are not appropriate for risk management where simulation and stress testing on tranche values are required.

For the purposes of this work, accuracy refers to the absolute or relative error of an approximate quantity (e.g. Higham, 1996).

A brief description of the first models used to analyse single-name credit risk is given. This is followed by a description of synthetic CDO tranches. The difference between the single-name and portfolio models is that default correlation is taken into account. Two distinct simulation-based models are discussed.

The disadvantage of simulation-based methods is that they are computationally expensive. Analytic approaches are then surveyed. The problem of calibrating models to market data is then explored. For this purpose, accuracy is considered to be the error between the fair value produced by the model and the market quote. Different distributional forms are then explored in order to fit the model to market observed tranche spreads. Finally, the hedging of synthetic CDO tranches is discussed.

The literature is presented in chronological order to highlight the improvement with respect to accuracy and computational efficiency, since 1997, in the determination of the prices of synthetic CDO tranches.

This chapter concludes with the contribution of this research to the literature.

2.1 Credit-risk models

For the modelling of credit risk two classes of models exist: ‘structural’ and ‘reduced form’. The structural approach originated with Merton (1974), who applied the equity framework of Black & Scholes (1973) to the valuation of credit risk. Merton’s framework is described in the literature as the ‘option-theoretic’ or the ‘firm-value’ approach. In this framework default occurs whenever the value of the firm’s assets falls below the value of the firm’s liabilities. Default is determinable only at the maturity date of the debt. Black & Cox (1976) extended the model by allowing for default to occur at any time up until the maturity date of the debt. The time of default is specified as the first moment when the value of the firm reaches a lower predetermined bankruptcy covenant value or threshold. Thus the problem is transformed into an application of the first-passage-time problem as described in Feller (1971, chapter XIV). Bielecki & Rutkowski (2002) categorise first-passage-time models into those with deterministic interest rates and those with stochastic interest rates. Through-out this research interest rates are assumed to be deterministic as in Finger (unpublished), Hull & White (2001) and Arvanitis & Gregory (2001).

In the reduced-form approach, the default event is directly modelled as an unexpected arrival; the relationship between the value of the firm’s assets and default is not considered explicitly. This class of models is attributed to Jarrow & Turnbull (1992, 1995) and Duffie & Singleton (1999).

Both types of models may be extended to the multivariate case. However, ‘hybrid models’, which are structural models calibrated to market-implied default probabilities are more readily accepted by financial market practitioners because of their mathematical tractability in higher dimensions. The default correlations are obtained from the asset correlations. Since asset returns are not directly observable, the common practice is to proxy asset correlations using equity correlations.

2.2 Default probability estimation

O'Kane & Turnbull (unpublished) and Arvanitis & Gregory (2001) describe a market standard method for the valuation of CDS contracts. Li (1998) utilises the market standard CDS valuation equation to 'bootstrap' the default probabilities from bond spreads or credit default swaps. In the review paper by Luo (2005) three different methods for bootstrapping these default probabilities are discussed. A term structure of default probabilities is a key input for the valuation of CDOs.

2.3 Simulation methods

In Gupton et al. (unpublished) a one-period horizon is considered. We assume that the default probability for each asset or entity in a portfolio is known. Each asset in the portfolio is assumed to follow a diffusion process. Default occurs when this process first crosses a certain threshold. The 'threshold' (Finger, unpublished) or 'default boundary' (Iscoe & Krenin, unpublished a) or 'default barrier' (Avellaneda & Zhu, 2001) is determined as the inverse cumulative normal distribution of the default probability. Over the single period, for each asset, a set of pseudo-random numbers is drawn from a standard normal distribution with zero mean and unit variance. Those that are smaller than the threshold value are deemed to have defaulted. This process is repeated to obtain the expected loss.

The multi-step method is described by Finger (unpublished), Hull & White (2001) and Arvanitis & Gregory (2001). In this approach a stepwise simulation is required. The pool of assets is simulated over the first period, the defaulted assets are tabulated, then another simulation is conducted for the next period and defaulted assets are tabulated again. This process is repeated over all the subsequent periods until the maturity of the contract. In addition default correlation is taken into account. A main numerical challenge in implementing the multi-step approach is the determination of the default thresholds. For a single period the default boundary is determined as follows. Let k_1 and p_1 denote the default threshold and the default probability for a single period respectively. Then the default threshold is given by

$$k_1 = \Phi^{-1}\left(p_1\right),$$

where Φ denotes the cumulative distribution function for the standard normal distribution.

Extending this to the multi-period setting, it is necessary to condition on the asset not defaulting until the beginning of the next period. This is best illustrated with the description of Finger (unpublished). Let k_t, $t = 1,..,T$, denote the default thresholds and T denotes the maturity date of the contract. Each company in the portfolio is modelled by a standard Brownian motion (where a rigorous definition is provided in section 4.1.2), $W_t^{(i)}$ with $W_0^{(i)} = 0$. The instantaneous correlation between distinct companies $W^{(i)}$ and $W^{(j)}$ is ρ. The company *i* defaults in the first period if $W_1^{(i)} < k_1$. For time $t > 1$, company *i* defaults in time period *t* if it has not defaulted during the first $t-1$ periods. Given that the probability of default in time period *t*, p_t is known (as these were extracted from CDS quotes), we have

$$P\left(W_1^{(i)} > k_1 \cap \ldots \cap W_{t-1}^{(i)} > k_{t-1} \cap W_t^{(i)} < k_t\right) = p_t .$$

To determine the thresholds k_t, $t = 2,..,T$, the first threshold is $\Phi^{-1}\left(p_1\right)$, numerical integration has to be performed on a *t*-variate cumulative normal distribution.

In determining these thresholds, Hull & White (2001) used an analytic approach to derive a recurrence relation for the integration. This requires numerical integration as the integral is not available in closed form. Arvanitis & Gregory (2001) stated that Monte Carlo simulation provides greater accuracy than the analytic method, although it is computationally slower. Avellaneda & Zhu (2001) considered a continuous-time version of Hull & White (2001). Iscoe & Krenin (unpublished a) used Monte Carlo simulation. In their approach the problem was reduced to the sequential estimation of the quantiles of the conditional distribution. Cheng (unpublished) compares the approaches of Avellaneda & Zhu (2001), Iscoe & Krenin (unpublished a) and a method that is a modified version of Zucca et al. (unpublished) but omits a direct comparison with Hull & White (2001) with respect to accuracy and computational efficiency. Cheng et al. (unpublished) reported long computational times and less accuracy than that of Iscoe & Krenin (unpublished a) in comparison to the scheme proposed by Cheng (unpublished).

For the pricing of CDOs two Monte Carlo simulations would then be required: one for determining the default boundary and another for determining the time of default. The major advantage of the multi-step method is that the asset value of the firm is simulated. The credit quality of the firm is then determined as the distance the simulated value is away from the default thresholds. The further away the firm value is the better the credit worthiness is. However, for the valuation of CDOs only the time of default is required, unless changes in credit quality are also required.

Li (2000) studied the survival time of each asset and defined the default correlation between two credit risks as the correlation coefficient between their survival times. A copula function is used to link the marginal distributions of survival times. This single-step approach produces default times directly and provides for greater computational efficiency than the multi-step approach albeit with slightly different results than the multi-step approach. This was analysed briefly in Morokoff (2003). Boscher & Ward (2002) applied Li (2000) to the valuation of CDO tranches. Standard and Poor’s have adopted Li (2000) as part of their rating methodology.

2.4 Semi-analytic methods

Monte Carlo methods are said to be slow for portfolios of assets larger than 100 names and suffer from numerical instability or variability when used to determine risk sensitivities (Greenberg et al, unpublished). There was an obvious need for an analytic method for pricing and risk management.

Vasicek (unpublished a, unpublished b) devised an analytic method based on dividing the individual risk of the company as a sum of two parts: a systemic and a specific risk component. This analytic form was based on there being entities in the portfolio with equal notional amounts and default probability.

The systemic risk factor is the risk related to the market or wider economy. The specific or idiosyncratic risk factor is related to the asset or entity under consideration. Conditional on the systemic factor being constant, the risks are independent, and this allows for the development of an analytic formula. In Vasicek (unpublished b) the analytic form developed was for the asymptotic case of homogeneous portfolios with infinitely many entities. This version of Vasicek's formula is used with a homogeneous correlation structure as in Greenberg et al. (unpublished).

However, in practice most CDO portfolios are heterogeneous. Laurent & Gregory (2003) extend Vasicek's approach by allowing for a heterogeneous portfolio by utilising the fast Fourier transform (FFT) to invert the characteristic function and reported significantly faster computational times with sufficient accuracy. Wang (1998) previously used copula functions and the FFT to determine aggregate loss distributions in the insurance context. Andersen et al. (2003) used a recursion technique and reported an improvement over the FFT approach. This approach would however be unsuitable for large portfolios (Klugman et al, 2004). An omission in the literature is to determine when the size of the portfolio becomes too large for recursion-based techniques to be computationally efficient relative to competing techniques. Bulhman (1984) establishes a limit when the use of the Panjer recursion (Panjer, 1981) for the compound Poisson distribution becomes uncompetitive with that of the FFT.

De Prisco et al. (2005) provided an analytic method based on the compound Poisson distribution to determine the tranche loss directly. This method differs from the other techniques in that the entire loss distribution is not calculated. The method used to determine the values of the compound Poisson distribution is the FFT. An omission is the use of the Panjer recursion technique. De Prisco et al. (2005) and Iscoe & Krenin (unpublished b) presented a method to assess computational efficiency and accuracy with respect to heterogeneity that is used in this research. Heterogeneity of the asset pool affects both accuracy and computational efficiency.

2.5 Calibration and risks of CDO tranches

The Gaussian copula has been used to determine implied correlations from the tranched Itraxx or iBoxx indices (Marshal et al, unpublished). This is similar to extracting the volatility using equity prices and the Black & Scholes (1973) model. The correlations implied by the observable prices of junior and senior tranches are generally higher than that needed to match the prices of mezzanine tranches, a phenomenon known as the 'correlation smile'. The main explanation for this phenomenon is the lack of tail dependence of the Gaussian copula (Kalemanova, unpublished).

Hull & White (2004) fit a so called ‘double *t*’ copula, when both the systemic and specific factors have a *t*-distribution, to observed market spreads. Kalemanova et al. (unpublished) proposed using the normal inverse Gaussian as a computationally more stable and efficient solution than the double-*t* distribution.

Gibson (unpublished) demonstrated by the use of a single factor Gaussian copula the amount of leverage in CDO tranches and the risks of CDO tranches to the business cycle.

2.6 Actuarial science and credit risk

The life-assurance and general-insurance branches of actuarial science are concerned with the losses incurred when an event such as death or accident materialises. In credit risk, the main event that triggers a loss is the default of the company or entity. This could be the non-payment of a coupon payment on a bond or loan. Li (2006) highlighted the prior and potential contribution of actuarial science to the field of credit risk:

> The modelling of default events using survival time distribution is very similar to the modelling of the death of a human life. A credit curve, which describes the term structures of default probabilities for an obligor, is very much like a mortality table. Pricing a default swap is not much different from pricing a life insurance contract. Some delicate issues, such as premium accrual or refund, have been well studied in basic actuarial textbooks, while some popular models like the one in Bloomberg are still based on some simple assumptions on the timing of default. The current popular credit portfolio model using copula functions (Li, 2000) was borrowed from the actuarial work of pricing joint-life annuity [sic] by considering the phenomenon of ‘‘broken heart’’ (Frees, Carriere, and Valdez, 1996). All efficient computational methods of a credit portfolio loss distribution under the framework of copula functions, such as the Fourier transformation, recursion or conditional normal approximation, have all been ‘‘reinvented’’ by credit derivative practitioners while they had been extensively studied by actuaries much earlier. Duffie and Singleton’s approach (1999) to risky bond pricing can be traced back to the recursive formulas for insurance reserving by the Hattendorf theorem (1868). In fact, the whole reserve issue for illiquid products, and various products involving extreme events or ‘‘gap risk’’ in credit derivatives, lacks fundamental thinking and rigor

> compared to what has been achieved in the actuarial science area for similar risks . One of the most popular credit portfolio models, CreditRisk+ (Wilde, unpublished), was all based on actuarial work of the compound Poisson distribution approach to loss modelling, and most of the formulas in CreditRisk+ are taken directly from the actuarial book by Panjer and Willmot (1992).In summary, credit derivatives are becoming the ''bread and butter'' of the credit market. Actuaries with a long tradition and experience in modelling similar events should be able to make more of a contribution to the area, and life insurers, as large investors in the credit space, should be more active users of the technology. We are looking forward to more research papers that cover credit modelling in the future in NAAJ.

2.7 Contribution of this research

They are two categories or models considered in this research; the Monte Carlo and the semi-analytic approaches. The Monte Carlo approaches implemented are:

- the multi-step approaches of Finger (unpublished), Arvanitis & Gregory (2001) and Hull & White (2001) as well as three methods to determine the default thresholds; and
- the default-time approach (Li, 2000).

The semi-analytic approaches implemented are:

- the numerical inversion of the characteristic function (Laurant & Gregory, 2003);
- the direct convolution technique via recursion (Andersen et al, 2003);
- the Panjer recursion (Panjer, 1981) approximation to the compound Poisson distribution; and
- the FFT approximation to the compound Poisson distribution process (De Prisco et al, 2005).

What has not been addressed in the literature is an overall comparison of the performance of the approaches identified above with respect to accuracy and computational efficiency using heterogeneous portfolios.

Another omission that is rectified in this research is a guide to the implementation of the models. Typically, the published works omit the algorithms.

The role of techniques that are typically used in actuarial science is highlighted.

This research fills the voids identified in the literature. The differences in the models are identified and explained. The values obtained using the different techniques are compared. The advantages and disadvantages of the models are discussed.

In addition, using semi-analytic techniques to value CDOs, the impact of heterogeneity of CDO portfolios is assessed.

2.8 Conclusion

The review dealt with all the major models used for the valuation of synthetic CDO tranches since the inception of trading. In the next chapter, the valuation of a CDS is described and critically a method to extract default probabilities from market quotes of CDSs is provided.

Chapter 3

Credit Default Swaps

CDSs play a critical role in the valuation of CDOs. They are primarily used to extract risk-neutral default probabilities of companies. These default probabilities are used in the valuation of market value of CDO tranches, which is the main purpose of this research. In this chapter, a valuation method for credit default swaps is given. The process of obtaining the risk-neutral default probabilities from CDSs quotes is demonstrated. The chapter concludes with the assumption of a constant or ‘flat’ CDS spread and how this simplifies default-probability determination.

3.1 Hazard rate

Li (1998) provides a method to extract default probabilities from bond spreads. Arvanitis & Gregory (2001) extend the approach to extracting default probabilities from CDS spreads. Let *T* denote a continuous random variable measuring the length of time until default for any security. The distribution function *F,* is denoted by

$$F(t) = P(T \le t),\ t \ge 0,$$

where *P* is the density function, and set

$$S(t) = 1 - F(t) = P(T > t),\ t \ge 0,$$

where the function $S(t)$ is the survival function. When $t = 0,\ S(0) = 1.$ The probability density function of the default time is defined as

$$f(t) = F^{'}(t) = -S^{'}(t) = \lim_{\Delta t \to 0^{+}} \frac{P(t < T \le t + \Delta t)}{\Delta t}.$$

The hazard rate function gives the security’s default probability over the time interval $[t, t + \Delta t]$ if it has survived to time *x*. It is defined as,

$$P(t < T \le t + \Delta t | T > t) = \frac{F(x + \Delta t) - F(x)}{1 - F(x)} \approx \frac{f(t)\Delta t}{1 - F(x)}.$$

The hazard rate function

$$h(x)=\frac{f(x)}{1-F(x)},$$

has a conditional probability interpretation as the probability density function of default at exact age x, given survival to that time. In actuarial science, the hazard rate is referred to as the force of mortality.

The hazard rate function, the distribution function and the survival function are related by

$$h(x)=\frac{f(x)}{1-F(x)}=-\frac{S'(x)}{S(x)}.$$

The survival function can then be expressed in terms of the hazard rate function using

$$S(t)=\exp\left[-\int_0^t h(s)ds\right].$$

A common assumption is to assume that the hazard rate is constant over certain periods, for example $[x,x+1]$. In this case the probability density function of survival time of a security is

$$f(t)=a\exp(-at).$$

Bluhm et al. (2003) remark that the hazard rate $h(t)$ may be regarded as the 'forward default rate'. It is analogous to the forward interest rate with zero-coupon bond prices corresponding to survival probabilities.

3.2 Valuation of credit default swaps

Once the hazard rate is specified, the probability of default occurring before time t is given by

$$F(t)=1-\exp\left(-\int_0^t h(u)du\right).$$

The premium leg of a CDS is defined as a fixed series of cash flows at times $t_1,t_2,\ldots,t_n$, which are paid until $t_n=T$ or until default. Let $B(t_i)$ denote the risk-free discounting factor or the price of a risk-free zero-coupon bond maturing at time t_i and let X_T denote the CDS premium. Unlike other insurance markets, the premium is paid in arrears. In the event of default the protection buyer pays the protection seller the accrued premium. The determination of which is discussed in the next section. The discounted expected value under the risk neutral measure of the premium leg is

$$P = \sum_{i=1}^{n} \left(t_i - t_{i-1}\right) X_T B\left(t_i\right) S\left(t\right). \tag{3.1}$$

Similarly, the discounted expected value under the risk-neutral measure of the default leg is

$$D = \left(1 - R\right) \int_{0}^{T} h\left(s\right) \, S\left(s\right) B\left(s\right) ds \tag{3.2}$$

where R is the recovery rate (Arvanitis & Gregory, 2001).

This represents the expectation over all possible default times. It is possible to evaluate this expression via numerical integration which is computationally intensive. A simpler alternative is to value it approximately by evaluating it at m dates; hence the default leg is given by

$$D = (1 - R) \sum_{j=1}^{m} \left[F\left(t_j\right) - F\left(t_{j-1}\right) \right] B\left(t_j\right). \tag{3.3}$$

For the premium leg, the interval used is the time between premium payment dates. For the default leg, the time interval $\left[t_{j-1}, t_j\right)$ should be as small as possible, to approximate the possibility that default can occur at any time. Arvanitis & Gregory (2001) use a time-step of 1 day for the default leg, whereas O'Kane & Turnbull (unpublished) used a monthly time-step. A comparison of the accuracy of the monthly versus daily discretisation is given in section 3.4. For an n-year contract the number of daily time-steps would thus be $m = 365n$.

At equilibrium the premium leg and the default legs are should be equal. This is given by

$$P = D; \text{ and}$$

$$\sum_{i=1}^{n} \left(t_i - t_{i-1}\right) X_T B\left(t_i\right) S\left(t\right) = (1 - R) \sum_{j=1}^{m} \left[F\left(t_j\right) - F\left(t_{j-1}\right) \right] B\left(t_j\right).$$

This enables us to solve for the market premium of a default swap, X_T,

$$X_T = \frac{(1-R)\sum_{j=1}^{m}\left[F\left(t_j\right)-F\left(t_{j-1}\right)\right]B\left(t_j\right)}{\sum_{i=1}^{n}\left(t_i-t_{i-1}\right)\left(1-F\left(t_i\right)\right)B\left(t_i\right)}. \tag{3.4}$$

3.3 Accrued premium

It has become market practice to pay the default swap premiums in arrears. In equation (3.4) the accrued premium is not taken into account. For example, in the event of default at time τ, between premium payment dates t_k and t_{k+1}, the accrued premium would amount to

$$\frac{\tau - t_k}{t_{k+1}-t_k} X_T .$$

Arvanitis & Gregory (2001) derived the following approximation that should be added to the premium leg to take into account the accrued premium

$$A \approx \sum_{j=1}^{m}\left[F\left(t_j\right)-F\left(t_{j-1}\right)\right]\frac{\left(t_j - t_{j^*-1}\right)X_T}{\left(t_{j^*}-t_{j^*-1}\right)}B\left(t_j\right) \tag{3.5}$$

where t_{j^*} represents the next premium payment date after t_j and t_{j^*-1}, the last premium date after t_{j-1}. Note that the interval $\left[t_{j-1}, t_j\right)$ is smaller than the interval between premium payment dates, t_{j^*} is the number rounded towards the next premium payment date.

To determine the market premium which includes the accrued premium, denoted by X_T^{Acc}, we use the fact that at inception, the sum of the expected value of the discounted premiums and accrued premiums is equal to the sum of the expected value of the default losses. This is given by

$$P + A = D.$$

Substituting from equations (3.1), (3.2) and (3.5), we obtain:

$$\sum_{i=1}^{n}(t_i - t_{i-1})X_T^{Acc}\, B(t_i)S(t) +$$

$$\sum_{j=1}^{m}\left[F(t_j)-F(t_{j-1})\right]\frac{\left(t_j - t_{j^*-1}\right)X_T^{Acc}}{\left(t_{j^*}-t_{j^*-1}\right)}B(t_j) = (1-R)\sum_{j=1}^{m}\left[F(t_j)-F(t_{j-1})\right]B(t_j).$$

Solving for the market premium yields the following expression:

$$X_T^{Acc} = \frac{(1-R)\sum_{j=1}^{m}\left[F(t_j)-F(t_{j-1})\right]B(t_j)}{\sum_{i=1}^{n}(t_i - t_{i-1})B(t_i)\left(1-F(t_i)\right)+\sum_{j=1}^{m}\frac{\left[F(t_j)-F(t_{j-1})\right]\left(t_j - t_{j^*-1}\right)}{\left(t_{j^*}-t_{j^*-1}\right)}B(t_j)}. \tag{3.6}$$

3.4 Default-leg discretisation errors

The recommendation of Arvanitis & Gregory (2001) to use a time step of one day for the default leg is tested against that of O'Kane & Turnbull (unpublished) who recommended a monthly approximation. In Table 2.1, direct integration of equation (3.2) is compared with that obtained by the discrete approximation of equation (3.3). The hazard rates were assumed to be constant over a hypothetical ten-year CDS contract. In addition a continuous risk-free rate of 5% p.a. and recovery rate of 30% were assumed.

Table 3.1: Comparison of the default leg value using a daily and monthly approximation versus the continuous case

Hazard Rate (basis points)	Direct Integration	Daily (%)	Monthly (%)
50	2.69214	0.01	0.21
100	5.26386	0.01	0.21
200	10.0676	0.01	0.20
400	18.4623	0.01	0.21

The error for both the daily and monthly approximation is consistent over the range of values. This analysis confirms the conclusion of O'Kane & Turnbull (2003), that given the nearly 30-fold increase in computation necessary for the daily approximation, the monthly approximation is reasonable.

However, for the constant hazard case the direct integration of the default leg yields a closed form solution. Let a be the constant hazard rate, T the maturity of the contract, R the recovery rate and r the risk-free rate. Then the following result holds,

$$D = \frac{\left(1 - R\right) a \left(1 - e^{-\left(a+r\right)T}\right)}{a + r}.$$

3.5 Default probability determination

The observed default swap premiums reflect both the default probability and the recovery rate. It is not possible to extract both of these quantities from market quotes. A solution to this problem is to use the recovery rates based on studies by rating agencies (Hamilton et al, unpublished)**.** An appropriate value is in the range 10-50% (Arvanitis & Gregory, 2001).

3.5.1 Bootstrapping default probabilities

To price a CDS we need the cumulative default probabilities up to time *t,* $F\left(t\right)$. The standard modelling assumption used to price CDSs, is to assume that the hazard rate is a piecewise function of maturity time of the CDS (Arvanitis & Gregory, 2001 and O'Kane & Turnbull, unpublished). Assuming that the hazard rates are constant between M maturity dates, $[t_1, t_2, ..., t_M]$, the hazard rate function is given by

$$h\left(u\right) = h_k, \quad u \in \left[t_k - t_{k-1}\right], \ k = 1, ...M, \text{ where } t_0 = 0.$$

A combined hazard rate and time value function, $g\left(u\right)$, defined as follows, is useful for the numerical determination of the hazard rates. Where $g\left(u\right)$ is defined as,

$$g\left(u\right)=\begin{cases} h_1t_1 & \text{if } 0<u\leq t_1 \\ h_1t_1+h_1\left(u-t_1\right) & \text{if } t_1<u\leq t_2 \\ h_1t_1+h_2\left(t_2-t_1\right)+h_2\left(u-t_2\right) & \text{if } t_2<u\leq t_3 \\ \vdots & \vdots \\ h_1t_1+h_2\left(t_2-t_1\right)+\cdots+h_M\left(u-t_{M-1}\right) & \text{if } t_{M-1}<u\leq t_M\ . \end{cases}$$

Thus,

$$F\left(t_i\right)=1-e^{-g\left(t_i\right)}.$$

The hazard rates are "bootstrapped", solving for the hazard rate for the shortest period and then using this result to solve for the hazard rate for the next shortest period and so on. This procedure is identical to that used to determine the zero-coupon interest rate curve (Hull, 2003). In order to implement a bootstrapping algorithm to extract the default probabilities, it is useful to rewrite equation (3.6) in terms of survival probabilities with the piecewise constant hazard rates,

$$X_T=\frac{(1-R)\sum_{j=1}^{m}\left[e^{-g\left(t_{j-1}\right)}-e^{-g\left(t_j\right)}\right]B\left(t_j\right)}{\sum_{i=1}^{n}\left(t_i-t_{i-1}\right)B\left(t_i\right)e^{-g\left(t_i\right)}+\sum_{j=1}^{m}\left[e^{-h'\left(t_{j-1}\right)}-e^{-g\left(t_j\right)}\right]\frac{\left(t_j-t_{j^*-1}\right)X_T}{\left(t_{j^*}-t_{j^*-1}\right)}B\left(t_j\right)} \tag{3.7}$$

3.5.2 Bootstrapping example

The CDS quotes given in Table 3.2 (Arvanitis & Gregory, 2001) are to be used to obtain default probabilities. A recovery rate of 30% and semi-annual CDS premium payments are assumed.

Table 3.2: Data used in the calibration example

Maturity (years)	Market Premium (basis points p.a.)
0.5	50
1	60
3	80
5	105
7	120
10	140

The piecewise hazard rates are obtained by equating X_T in equation (3.7) to the market CDS premiums. This entails solving non-linear equations in order of increasing maturity. Mathematica and Matlab (which are proprietary numerical analysis software packages), have built-in utilities, FindRoot and fsolve respectively. Alternatively, the algorithm may be sourced from Kelly (1995), Quarteroni et al. (2007) or Yang (2005). The Matlab algorithm, newtons.m of Yang (2005) was used for solving non-linear equations (see Appendices).

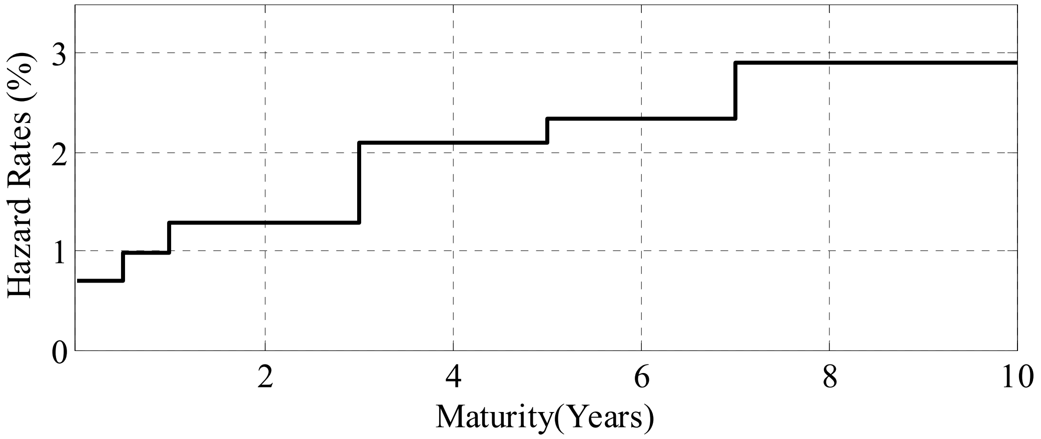

Figure 3.1 Bootstrapped hazard rates based on the data given in Table 3.2

Table 3.3: The results from the bootstrapping procedure for the CDS quotes in Table 3.1

Maturity (years)	Hazard rates (%)
0.5	0.704841
1	0.985117
3	1.27544
5	2.08629
7	2.34497
10	2.89385

3.6 Constant default swap spread case

An approximation for the hazard rate is derived in this section. Assuming that:

- the CDS premium is paid continuously;
- the protection amount is paid at the time of default;
- the recovery rate is constant;
- the CDS spread and hazard rate term structures are flat.

Then,

$$\begin{gathered} X_T \int_0^T e^{-(r+h)} ds = (1-R) \int_0^T h \exp\left(-\int_0^S h\, du \right) B(s)\, ds \\ \frac{X_T \left(1-e^{-(r+h)T}\right)}{r+h} = \frac{(1-R)h\left(1-e^{-(r+h)T}\right)}{r+h} \end{gathered} \tag{3.8}$$

$$h = \frac{X_T}{(1-R)} \tag{3.9}$$

The quality of this approximation is then checked by assuming that all the CDS spreads in Table 3.2 are constant for the full ten-year term. The approximated result is compared with that obtained using equation (3.7).

Table 3.4 The constant hazard approximated by the two methods.

CDS spreads (basis points)	Approximation by equation (3.5) (%)	Approximation by equation (3.7) (%)	Absolute deviation (%)
50	0.70	0.71	0.01
60	0.84	0.86	0.01
80	1.12	1.14	0.02
105	1.47	1.50	0.03
120	1.68	1.71	0.04
140	1.96	2.00	0.04

The approximation developed above appears to be close to the more accurate result obtained using equation (3.7). The maximum deviation is four basis points.

3.7 Conclusion

In this chapter the pricing of CDSs has been introduced. Approximations used have been explored and have been verified. The critical issue of extracting default probabilities from market quotes has been demonstrated. These default probabilities will be used to evaluate tranches of a CDO portfolio. That is the subject of subsequent chapters.

Chapter 4

Monte Carlo Methods Part I: The Multi-Step model

This chapter provides details on the first of two types of models considered in this research that use Monte Carlo simulation to determine the aggregate loss distribution over a multiple period. The other type of model, based on copula functions, is presented in the following chapter. In this chapter the multi-step model of Finger (unpublished), Arvanitis & Gregory (2001) and Hull & White (2001) is introduced. This model provides a realistic representation of the value of a firm and it is based on the single-period structural model of Merton (1974). Default occurs when the value of the firm crosses a default boundary or threshold, which represents its liabilities. The determination of the default thresholds is a computationally intensive process. Iscoe & Krenin (unpublished a), Hull & White (2001) and Morokoff (2003) propose different methods to determine the default boundaries. The literature omits an exposition of the implementation of these methods and a direct comparison. In the last section a four-tranche heterogeneous synthetic CDO is evaluated.

4.1 Mathematical review

In this section a brief review of the essential mathematics used in this chapter is provided.

4.1.1 Probability space

A probability space is the triplet $(\Omega,\omega,\mathbb{P})$ where Ω is a sample space, ω is a collection (assumed to be an algebra) of events (each a subset of Ω), and $\mathbb{P}$ is a probability function with domain ω (Mood et al, 1974, p.25).

4.1.2 Brownian motion and geometric Brownian motion

A standard one-dimensional Brownian motion on $[0,T]$, is a stochastic process $\{W(t), 0 \le t \le T\}$ with the following properties:

i. $W(0)=0$;

ii. $W(t)$ is a continuous function on $[0,T]$;

iii. the increments $\{W(t_1)-W(t_0), W(t_2)-W(t_1), ..., W(t_k)-W(t_{k-1})\}$ are independent for any k and for any $0 \leq t_0 < t_1 < ... < t_k \leq T$;

iv. $W(t)-W(s) \sim N(0,t-s)$ for any $0 \leq s < t \leq T$.

A consequence of (i) and (iv) is that

$$W(t) \sim N(0,t)$$

for $0 < t \leq T$ (Glasserman, 2004, p79; Baxter & Rennie, 1996, p.48). Brownian motion is also called a Wiener process (Baxter & Rennie, 1996, p.50).

Let $Z_1,...,Z_n$ be independent standard normal random variables. For a standard Brownian motion, set $t_0 = 0$ and $W(0) = 0$. Subsequent values can be generated as follows:

$$W(t_{i+1}) = W(t_i) + \sqrt{t_{i+1} - t_i} Z_{i+1,} \quad i = 0,...,n-1.$$

The main deficiency of using a standard Brownian motion for asset price values is captured by Samuelson (1965), who also proposes a solution:

> The anomaly apparently came because Bachelier had forgotten that stocks possess limited liability and thus cannot become negative, as is implied by the arithmetic Brownian process. To correct this, I introduced the "geometric" or "economic Brownian motion," with the property that every dollar of market value is subject to the same multiplicational or percentage fluctuations per unit time regardless of the absolute price of the stock.

For geometric Brownian motion the percentage changes

$$\frac{S(t_2)-S(t_1)}{S(t_1)}, \frac{S(t_3)-S(t_2)}{S(t_2)}, ..., \frac{S(t_n)-S(t_{n-1})}{S(t_{n-1})}$$

are independent for $t_1 < t_2 < ... < t_n$ rather than the absolute changes $S(t_{i+1}) - S(t_i)$.

4.1.3 The multivariate normal distribution

A multivariate normal distribution $N(\mu,\Sigma)$ is specified by its mean μ and covariance matrix Σ. The covariance matrix may be specified through its diagonal entries σ_i^2 and correlations ρ_{ij}. The joint probability density function (pdf) of $(x_1,...,x_N)$ is given by:

$$f_X(x_1,...,x_N) = \frac{1}{(2\pi)^{N/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right).$$

4.1.4 Generating correlated multivariate normal random variates

The algorithm to generate random correlated variables according to a multivariate normal distribution with the vector of expected value μ and covariance matrix Σ is:

Step 1

Obtain the Cholesky factor for Σ, i.e., an upper triangle matrix such that $\Sigma = U^T U$.

Step2

Generate n independent standard normal variates $Z_1,...,Z_n \sim N(0,1)$.

Step 3

$$X = \mu + U^T Z, \text{ where } Z = \left[Z_1,...,Z_n\right]^T$$ (Brandimarte, 2006).

4.1.5 Discrete Fourier transforms

The discrete Fourier and the FFT are discussed. A discrete Fourier transform (DFT) transforms one function to another. The main use of Fourier transforms in this research is to gain computational tractability via the use of the FFT. The FFT is a computationally efficient implementation of the DFT. The FFT has been used in diverse fields such as the enhancement of digital photographs (Bracewell, 2000, p. 281). The discussion in this section is based on Cerny (2004, p.164).

4.1.5.1 The discrete Fourier transform

Let $z_n = e^{\frac{i2\pi}{n}}$ and $a_0, a_1, \ldots, a_{n-1}$ be a sequence of n numbers. The DFT of $a_0, a_1, \ldots, a_{n-1}$ is the sequence $b_0, b_1, \ldots, b_{n-1}$ such that:

$$\begin{aligned} b_k &= \frac{a_0\left(z_n^k\right)^0 + a_1\left(z_n^k\right)^1 + \ldots + a_{n-1}\left(z_n^k\right)^{n-1}}{\sqrt{n}} \\ &= \frac{1}{\sqrt{n}} \sum_{j=0}^{n-1} a_j e^{\frac{i2\pi}{n} jk} \\ &= \frac{1}{\sqrt{n}} \sum_{j=0}^{n-1} a_j z_n^{jk}. \end{aligned}$$

This is commonly written as

$$\mathcal{F}\left(a\right) = b.$$

The above is referred to as the forward Fourier transform. The inverse Fourier transform is

$$\begin{aligned} \tilde{a}_l &= \frac{\tilde{b}_0\left(z_n^{-l}\right)^0 + \tilde{b}_1\left(z_n^{-l}\right)^1 + \ldots + \tilde{b}_{n-1}\left(z_n^{-l}\right)^{n-1}}{\sqrt{n}} \\ &= \frac{1}{\sqrt{n}} \sum_{k=0}^{n-1} \tilde{b}_k z_n^{-kl} \\ &= \frac{1}{\sqrt{n}} \sum_{k=0}^{n-1} \tilde{b}_k e^{-i\frac{2\pi}{n} kl}, \end{aligned}$$

which may be written as

$$\tilde{a} = \mathcal{F}^{-1}\left(\tilde{b}\right).$$

4.1.5.2 The FFT

Let $n = 2^p$. The Fourier transform is as given above

$$\begin{aligned} b_k &= \frac{1}{\sqrt{n}}\sum_{j=0}^{n-1} a_j \left(z_n^k\right)^j \\ &= \frac{1}{\sqrt{n}}\sum_{j=0}^{(n/2)-1} a_{2j}\left(z_n^k\right)^{2j} + \frac{1}{\sqrt{n}}\sum_{j=0}^{(n/2)-1} a_{2j+1}\left(z_n^k\right)^{2j+1} \\ &= \frac{1}{\sqrt{n}}\sum_{j=0}^{(n/2)-1} a_{2j}\left(\left(z_n^2\right)^k\right)^j + \frac{1}{\sqrt{n}} z_n^k \sum_{j=0}^{(n/2)-1} a_{2j+1}\left(\left(z_n^2\right)^k\right)^j . \end{aligned}$$

The two sums represent transforms of length $n/2$. The above could be reused to compute $b_{k+n/2}$ and noting that

$$\begin{aligned} z_n^{\frac{n}{2}} &= \left(e^{\frac{i2\pi}{n}}\right)^{\frac{n}{2}} \\ &= e^{i\pi} \\ &= -1 \end{aligned}$$

and

$$\begin{aligned} z_n^n &= \left(z_n^{\frac{n}{2}}\right)^2 \\ &= 1 \end{aligned}$$

which follows from Euler's formula.[1] Hence,

$$\begin{aligned} b_{k+n/2} &= \frac{1}{\sqrt{n}} z_n^n \sum_{j=0}^{(n/2)-1} a_{2j}\left(\left(z_n^2\right)^k\right)^j + \frac{1}{\sqrt{n}} z_n^{k+n/2} \sum_{j=0}^{(n/2)-1} a_{2j+1}\left(\left(z_n^2\right)^k\right)^j \\ &= \frac{1}{\sqrt{n}}\sum_{j=0}^{(n/2)-1} a_{2j}\left(\left(z_n^2\right)^k\right)^j - \frac{1}{\sqrt{n}} z_n^k \sum_{j=0}^{(n/2)-1} a_{2j+1}\left(\left(z_n^2\right)^k\right)^j . \end{aligned}$$

The sums

$$\sum_{j=0}^{(n/2)-1} a_{2j}\left(\left(z_n^2\right)^k\right)^j \text{ and } \sum_{j=0}^{(n/2)-1} a_{2j+1}\left(\left(z_n^2\right)^k\right)^j$$

need to be computed for $k = 0,1,...,n/2-1$, the values for $k = n/2$ are the same with $k = 0,$ as is the value for $n/2+1$ and $k = 1$.

[1] Euler's formula $e^{ix} = \cos x + i\sin x$ holds for any real x . When $x = \pi$, we have $\cos\pi + i\sin\pi = -1$. Euler's identity follows from the formula: $e^{i\pi} + 1 = 0.$

These two sums for different k represent two Fourier transforms of length $n/2$. To obtain b_k for $k = 0,1,...,n-1$ the only extra computation required is n complex multiplications by the factor z_n^k. A transform of length n requires two transforms of length $n/2$ plus n complex multiplications. This is $n/2$ transforms of length two. The algorithm is applied recursively.

4.1.5.3 The advantages of the FFT

Let $O(n)$ denote the number of complex multiplications required for a transform of size n and $O(1) = 1$. Then

$$\begin{aligned}
O(n) &= 2O(n/2) + n \\
&= 2\left(2O(n/2) + n/2\right) + n \\
&= 4O(n/4) + n + n \\
&= 2^p O\left(n/2^p\right) + pn \quad \text{(after } p \text{ steps)} \\
&= 2^p O(1) + pn \quad \text{(since } n = 2^p\text{)} \\
&= pn + n \qquad (O(1)=1) \\
&= n\log_2 n + n.
\end{aligned}$$

The discrete Fourier transform is quadratic i.e. $O(n) = n^2$, (Press et al, 1997, p. 504), and when this is compared to the relationship derived above, a significant reduction in computational time is to be observed.

4.2 Single-period models

The Merton (1974) model is generally given as the starting point for the modelling of credit risk. The concept of default threshold is illustrated for a single-entity portfolio. Merton allows for default only at the end of the contract. The asset value is modelled as a geometric Brownian motion. Default occurs when the asset value falls below a threshold. Creditmetrics (Gupton et al, unpublished) is based on Merton (1974) and it determines the default threshold, k, for an asset with default probability p as:

$$k = \Phi^{-1}(p),$$

where Φ^{-1} is the inverse of the normal cumulative distribution function with zero mean and unit variance.

4.2.1 Portfolio credit risk

When modelling portfolio credit losses it is vital to incorporate default correlation. This is because the changes in credit quality of the assets in the portfolio are affected in part by the same macro-economic variables (Gupton et al, unpublished, p.36).

The entire correlation structure among binary variables is not completely specified by the pairwise correlations. For example, consider a three-asset portfolio with default probabilities p_1, p_2 and p_3. The correlation matrix will give the pairwise correlations ρ_{12}, ρ_{13} and ρ_{23} that allow for the calculation of the probability of default for any two assets (Arvanitis & Gregory, 2001). The ability to calculate the probability that all assets default is restricted by lack of information regarding the joint default probability p_{123}. Hence, Arvanitis & Gregory (2001) assume that the default structure can be determined from the multivariate normal distribution. A key property of the multivariate normal distribution is that the correlation structure is defined by the covariance matrix; hence all default probabilities are uniquely determined by the pairwise correlations. The assumption of using the multivariate normal is implicit in Gupton et al. (unpublished).

4.2.2 Generating the loss distribution

Consider an n-asset portfolio $\tilde{X} = (X_1, ..., X_n)$. Each asset has a default probability, $p_i,\ i = 1, ..., n,$ and common default correlation Λ. In this case we assume that the recovery rate is zero. The following algorithm determines by simulation the number of defaults that occur and hence determines the loss distribution over a single period:

Step 1

Determine the default thresholds for each asset $k_i = \Phi^{-1}(p_i) \quad i = 1, ..., n.$ Denote the vector of thresholds as $\tilde{k}$. Determine the Cholesky factorization matrix of the correlation matrix A such that $\Lambda = AA',$ where A is a triangular matrix and A' is its transpose. The cumulative loss to zero, $L = 0.$

Step 2

For each simulation run, generate the vector of n independent random normal variates, $Y_i, i = 1,...,n,$, and multiply this vector by A. This yields a vector of correlated random normal variates, $\tilde{Y} = AY$ with elements $\tilde{y}_i, i = 1,\ldots,n.$

Step 3

If any $\tilde{y}_i \leq k_i$, then default has occurred. Update the cumulative loss total to reflect any losses:

$$L = L + \sum_{i=1}^{n} X_i \mathbf{1}_{\{\tilde{y}_i \leq k_i\}}.$$

Repeat steps 2 and 3.

Step 4

The mean and standard deviation of the cumulative loss distribution can now be determined.

A similar approach for generating the loss distribution was used by the Basel Committee on Banking Supervision (unpublished) in their Asymptotic Single Risk Factor (ASRF) formula. This formula, which was developed by Gordy (2003) is used to determine regulatory capital. Whilst the approach outlined above can be used for a portfolio with assets having notional amounts of various sizes, the ASRF formula is useful only in cases where each asset's nominal amount in a portfolio is approximately equal. Lütkebohmert (chapter 4, 2009) contains a detailed discussion of the ASRF formula.

4.3 The multi-step model

The importance of multi-step models beyond its use in credit derivatives was highlighted by McLeod et al. (2005), who found that those market participants that use a single-period model are potentially overestimating their credit risk.

If a high rate of default is experienced in one period, then it is likely that those corporate names or ‘obligors’ that did not default would have generally decreased in credit quality. The effect would then be that the default rate for the second period would also be high (Finger, unpublished). The model considered in this section aims to capture this phenomenon.

The first model to analyse portfolio credit risk over multiple periods is attributed to Iscoe et al. (unpublished). Their approach is a multifactor approach, similar to that of Moodys (Bluhm et al, 2003). The approach by Arvanitis & Gregory (2001) that is similar to the Creditmetrics (Gupton at al, unpublished) model extended to multiple periods is illustrated in this section. This model is identical to the ‘diffusion-driven Creditmetrics extension’ of Finger (unpublished). Hull & White (2001) developed their similar model independently (Arvanitis & Gregory, 2001, p158). These models have the same default thresholds. The differences between these models relate to the determination of the default thresholds. This is discussed in the next section. The firm value is treated as a stochastic quantity that results in a default event when the firm value is less than a certain threshold. For the exposition of the multi-step model, the following notation is used:

T, the maturity date for the transaction;

$V_t^{(i)}$, the *i*th firm value at time *t;*

n, the number of firms in the portfolio, *i*=1,…,*n;*

$\mu^{(i)}$, the mean rate of return of the *i*th firm;

$\sigma^{(i)}$, the volatility of the *i*th firm value;

$\Delta W_t^{(i)}$, multivariate standard normal random variable of the *i*th firm in the *t*th time interval, defined by Λ, $\Delta W_t^{(i)} = W^{(i)}\left(t_j\right) - W^{(i)}\left(t_{j-1}\right) \sim N\left(0, t_j - t_{j-1}\right)$;

k_j^i, the threshold in the interval $\left[t_{j-1}, t_j\right)$, for the *i*th firm value; the threshold is assumed to be constant over the interval.

We can now write the process for each firm value, $V_t^{(i)}$ of each underlying of the basket:

$$\frac{\Delta V_t^{(i)}}{V_t^{(i)}} = \mu^{(i)}\Delta t + \sigma^{(i)}\Delta W_t^{(i)} . \tag{4.1}$$

The default probabilities are bootstrapped from default swap quotes or bond spreads as described in chapter 3. Since the model is calibrated to the individual default probabilities, the firm's capital structure and its evolution according to equation (4.1), is no longer needed. Changing the mean or the variance of the rate of return, or both, will change the position of the calibrated threshold but will not change the default probability. In a footnote remark on this point Arvanitis & Gregory (2001, p.181) stated:

> This point may seem unusual. However, it can be readily checked in a one-period model that the parameters for two Brownian motions do not change the joint default probability when the model is calibrated to each individual default probability.

This is best illustrated with an example.

Example

Consider two Brownian motions over a single period, with default probabilities 5% and 6% and correlation 50%. The joint distribution is a bivariate normal. The default threshold for each Brownian motion over this single period is the inverse of the (univariate) normal distribution. Two cases are considered of the joint default probability of a standardised and non-standard bivariate normal distribution.

Case 1

The standardised bivariate normal distribution has the following parameters:

$$\begin{array}{lll} \mu^{(1)} = 0 & \sigma^{(1)} = 1 & \multirow{2}{*}{$\Lambda = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix},$} \\ \mu^{(2)} = 0 & \sigma^{(2)} = 1 & \end{array}$$

where Λ is the covariance matrix.

The individual default thresholds are:

$$\begin{aligned} k_1^1 &= \Phi^{-1}\left(0.05\right) = -1.64 \\ k_1^2 &= \Phi^{-1}\left(0.06\right) = -1.55. \end{aligned}$$

The joint cumulative distribution function $F\left(k_1^1, k_1^2\right) = 0.014$.

Case 2

In this case a non-standardised bivariate normal distribution is used, that has the following parameters:

$$\mu^{(1)} = 0.02 \qquad \sigma^{(1)} = 0.5 \qquad \Lambda = \begin{bmatrix} 0.25 & 0.0625 \\ 0.0625 & 0.0625 \end{bmatrix}.$$
$$\mu^{(2)} = 0.04 \qquad \sigma^{(2)} = 0.25$$

The individual default thresholds are:

$$k_1^1 = N_1^{-1}\left(0.05\right) = -0.80$$
$$k_1^2 = N_2^{-1}\left(0.06\right) = -0.35,$$

where N_1 and N_2 are normal distributions $N_1\left(0.02, 0.5\right)$ and $N_2\left(0.04, 0.25\right)$ respectively, N_1^{-1} and N_2^{-1} is the inverse normal distribution function of N_1 and N_2. Attention is drawn to the higher thresholds than that obtained in Case 1. The joint cumulative distribution function $F\left(k_2^1, k_2^2\right) = 0.014$, which is the same as in Case 1.

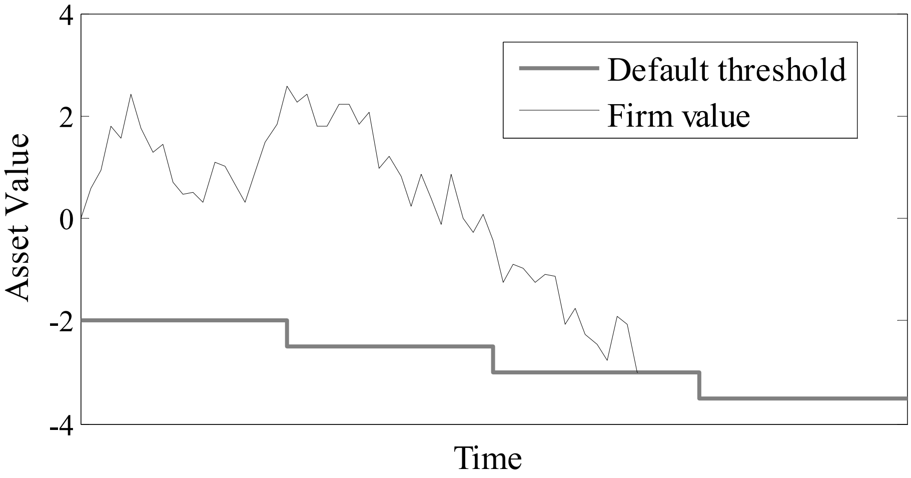

Figure 4.1: Evolution of the asset value of a single firm that eventually defaults

In Figure 4.1 the changes in asset value of a defaulting firm are illustrated. The evolution of the firms' asset value is illustrated. The time of default is when the asset value is below the default threshold.

We therefore take the mean to be zero and the variance to be equal to one, so that the return on the firm value, or its assets, is again represented by the standard normal variables. Assume that we have a series of S thresholds, which change over time. The time interval $\left[0,T\right]$ is divided into S time steps, $\left[t_0,t_1,...,t_S\right]$, where $t_0=0$ and $t_S=T$. At each time step, each firm value changes by $\Delta W_t^{(i)}$. The default probability in the sub period $\left[t_{j-1},t_j\right)$ at time j for the underlying i in the basket is denoted by $p_{j-1,j}^i$. The first threshold can be determined by:

$$k_1^i=\Phi^{-1}\left(p_{0,1}^i\right).$$

This ensures that

$$\mathrm{E}[V_1^i<k_1^i]=p_{0,1}^i,$$

i.e., the probability of the firm value being less than the threshold is equal to the probability of default. The next threshold has to be conditioned on the survival of the firm in the previous period. For example, the second threshold is determined so that

$$\mathrm{P}[V_1^i<k_1^i\cap V_2^i<k_2^i]=\mathrm{E}[1_{V_1^i>k_1^i}1_{V_2^i<k_2^i}]=p_{1,2}^i.$$

In general to find any threshold k_j^i, we apply the following formula:

$$\mathrm{E}\left[\left(\prod_{l=1}^{j-1}1_{\{k_l^i\geq V_l^i\}}\right)1_{\{k_l^i<V_l^i\}}\right]=p_{t_{j-1},t_j}^i. \tag{4.2}$$

The determination of the default thresholds is the subject to the next section. Default occurs for the ith firm in period d if for the first time the correlated Wiener process,

$$V_{t_d}^{(i)}<k_{d-1,d}^i.$$

The time of default τ_i of the ith firm is given by,

$$\tau_i=\min\left\{t_d\geq 0:V_{t_d}^{(i)}<k_d^i,d=0,1,...,S-1\right\}.$$

4.4 Default threshold determination

The default threshold is determined by solving for each k_j^i in equation (4.2). These thresholds may be determined by simulation or by semi-analytic formula (so called, as numerical methods are needed). In this section the approaches of Iscoe & Krenin (unpublished a), Morokoff (2003), Hull & White (2001) are investigated. For each approach the theory and the steps involved in its implementation are discussed, the computational time is recorded using a common portfolio and finally the most computationally efficient approach is identified.

4.4.1 Hull-White scheme

As before we have *n* companies and the time interval of the derivative contract is divided into *S* discrete intervals. The first step is to discretise the default probability densities so that defaults only happen at times $t_j \ \left(1 \leq j \leq n\right)$. The initial point is defined as $t_0 = 0$ and the time-step is given by $\delta_j = t_j - t_{j-1}$. Let q_j^i be the risk-neutral probability of default by a company *i* at time t_j and $V_j^{(i)}$ describe *i*th company's value at time t. Hull & White, (2001) define this as the credit worthiness index of the company *i.* It is assumed that $V_0^{(i)} = 0$ and that the risk-neutral process for $V_j^{(i)}$ is a Wiener process with zero drift and variance of 1 per year.

Let k_j^i be the value of the default barrier for a company at time t_j. Denote the risk-neutral probability density function for $V_j^{(i)}$ by f_j^i. The values of the variables k_j^i and f_j^i are determined inductively from the risk-neutral density probabilities q_j^i. Given the process for $V^{(i)}, V_1^{(i)} \sim N\left(0, \delta_1\right)$ is normally distributed with mean zero and a variance of δ_1. Then,

$$f_1^i\left(v\right) = \frac{1}{\sqrt{2\pi\delta_1}} e^{-\frac{v^2}{2\delta_1}},$$

and

$$q_1^i = \Phi\left(\frac{k_j^i}{\sqrt{\delta_1}}\right).$$

This implies that

$$k_1 = \sqrt{\delta_1}\Phi^{-1}\left(q_1^i\right).$$

For $2 \le j \le S$ we calculate k_j^i and f_j^i from q_j^i, k_{j-1}^i and f_{j-1}^i. The relationship between k_j^i and f_j^i is

$$q_j^i = \int_{k_{j-1}^i}^{\infty} f_{j-1}^i\left(u\right) N\left(\frac{k_j^i - u}{\sqrt{\delta_j}}\right) du. \tag{4.3}$$

The integral in the above equation is evaluated by numerical techniques for a given value of k_j^i. An iterative procedure can then be used to find the value of k_j^i that solves the equation. The density function f_j^i is

$$f_j^i\left(x\right) = \int_{k_j^i}^{\infty} f_{j-1}^i\left(u\right)\frac{1}{\sqrt{2\pi\delta_j}}\exp\left(\frac{-\left(v-u\right)^2}{\sqrt{\delta_j}}\right)du \tag{4.4}$$

for $v > k_j^i$. The equations (4.3) and (4.4) are solved numerically. Instead of direct numerical integration, Hull & White (2001) propose a discrete scheme. The details are provided below.

For each company we consider M values of $V_j^{(i)}$ between k_j^i and $5\sqrt{t_j}$ where M is several hundred. Let v_{jm} be the mth value of $V_j^{(i)}$ $\left(1 \le m \le M\right)$ and π_{jm} the probability that $V_j^{(i)} = v_{jm}$ with no earlier default. The discrete versions of equations (4.3) and (4.4) are

$$q_j = \sum_{m=1}^{M} \pi_{j-1,m}\,\Phi\left(\frac{k_j^i - v_{j-1,m}}{\sqrt{\delta_j}}\right)$$

and

$$\pi_{j,m} = \sum_{m=1}^{M} \pi_{j-1,m} p_{jmn},$$

where p_{jmn} is the probability that $V_j^{(i)}$ moves from $v_{j-1,m}$ at time t_{j-1} to v_{jn} at time t_j. We set

$$p_{jmn} = N\left[\frac{0.5\left(v_{ijn} + v_{i,j,n+1}\right) - v_{i-1,j,m}}{\sqrt{\delta_i}}\right] - N\left[\frac{0.5\left(v_{jn} + v_{j,n-1}\right) - v_{j-1,m}}{\sqrt{\delta_j}}\right]$$

where $1 < n < M$. When $n = M$,

$$p_{jmn} = 1 - N\left[\frac{0.5\left(v_{jn} + v_{j-1,n-1}\right) - v_{j-1,m}}{\sqrt{\delta_j}}\right]$$

and when $n = 1$,

$$p_{jmn} = N\left[\frac{0.5\left(v_{jn} + v_{j,n+1}\right) - v_{j-1,j,m}}{\sqrt{\delta_j}}\right] - N\left[\frac{k_j - v_{j-1,m}}{\sqrt{\delta_j}}\right].$$

The implementation is the direct application of the above formula. A Newton-Raphson scheme is used to solve for the thresholds.

4.4.2 Default threshold problem

Iscoe & Krenin (unpublished a) reduced the problem of default boundary estimation to a sequential estimation of conditional default probabilities.

4.4.2.1 Isoce & Kreinin model

For a given random process $\eta_t\left(t > 0\right)$, the inverse problem is formulated as follows: given a known probability distribution function, $P\left(t\right)$, find a boundary, $b\left(t\right)$, such that the first hitting time $\tau = \inf_{t>0}\left\{t : \eta_{\ t} < b\left(t\right)\right\}$ satisfies the equation

$$P\left\{\tau \le t\right\} = P\left(t\right), \quad t > 0.$$

Consider a discrete-time, mean-zero process, $S_n, n = 0,1,2,...,S_0$, having a finite variance σ_n^2 at time t_n. The standard process taking the value η_n at time t_n,

$$\eta_n = \frac{S_n}{\sigma_n}, \quad n = 1,2,...; \eta_0 = 0$$

satisfies the relation

$$\mathbb{E}\left[\eta_n\right]=0$$
$$\mathbb{E}\left[\eta_n^2\right]=1, \quad n=1,2,...$$

Assuming that S has independent increments, the process η_n is determined by

$$\eta_n = \frac{1}{\sqrt{\sigma_n}}\sum_{i=1}^{n}\varepsilon_i, \quad n=1,2,... \tag{4.5}$$

where ε_i is an independent random variable. A particular case of interest, which is considered below, arises when $\varepsilon_i \sim \mathrm{N}\left(0,\Delta t_i\right)$, $\Delta t_i = t_i - t_{i-1}$. Iscoe & Krenin (unpublished a) state the problem of boundary determination formally as follows.

The Default Boundary Problem.

Suppose that a sequence $P\left(n\right)$, $n=1,2,...,N$ is given and satisfies the conditions

$$0=P\left(0\right)\le P\left(1\right)\le P\left(2\right)\le ...\le P\left(N\right)<1$$

Let η_n, $n=1,...,N$ be defined as in equation (4.5). The boundary $k_1,k_2,...,k_N$ is determined so that the distribution $P\left(\tau\le t_n\right)$, $n=1,2,...,N$ of the first crossing time τ, satisfies the relation

$$P\left(\tau\le t_n\right)=P\left(n\right), \quad n=1,2,...,N$$

so that the sequence $P\left(n\right)$ describes the cumulative probability of default.

4.4.2.2 Computation of default probability

The algorithm for computing the default boundary is obtained via the sequential estimation of quantiles of the conditional default distributions.

Let $\pi_n = P\left(\tau = t_n\right)$ and $\hat{P}_n = P\left(\tau = t_n \middle| \tau \ge t_n\right)$ and consider the discrete-time process η_n $\left(n=1,2,...\right)$ defined as in equation (4.5).

Lemma

The boundary b_k, $k=1,...,N$ satisfies the equation

$$\pi_n = P\left\{\bigcap_{t=1}^{n-1}\{\eta_t \geq k_t\}, \eta_n < k_n\right\}, n = 1,2,...,N.$$

The probabilities π_n and $P(n)$ satisfy the following equations:

$$\pi_n = P(n) - P(n-1),\ n = 1,2,3...,N;$$

$$\hat{P}_n = \frac{\pi_n}{1-P(n-1)},\quad n = 1,2,3,...,N.$$

The problem of this approach is that a large amount of data are stored in memory and this grows with each simulation.

4.4.2.3 Recursive Monte Carlo algorithm

The Iscoe & Krenin (unpublished a) method is implemented as follows.

Step 1

The first threshold is determined analytically, from the relation

$$P(1) = P\{\eta_n \leq k_1\}.$$

If the cumulative distribution function for the first threshold is a normal distribution and is denoted by Φ, then b_1 is obtained from

$$k_1 = \Phi^{-1}(P(1)).$$

This was also the case for the Hull & White method.

Step 2

Compute conditional probabilities

$$\hat{P}_n = \frac{\pi_n}{1-P(n-1)},\quad n = 2,...,N.$$

The conditional default probabilities are required in Step 4.

For this research constant hazard rates have been assumed at equidistant valuation dates. In this case the above formula reduces to

$$\begin{aligned}\hat{P}_n &= \frac{P(n)-P(n-1)}{1-P(n-1)} \\ &= \frac{1-e^{-hn}-\left(1-e^{-h(n-1)}\right)}{1-\left(1-e^{-h(n-1)}\right)} \\ &= \frac{e^{-h(n-1)}-e^{-hn}}{e^{-h(n-1)}} \\ &= 1-e^{-h},\end{aligned}$$

and hence is constant for equidistant valuation dates.

Step 3

Suppose that the default boundary k_t has already been computed for $t = 1,...,n-1$ and it is required to determine the value of k_n. Generate a large number, M, of independent and identically distributed sample paths of Wiener processes,

$$\eta(m) = \left(\eta_1(m), \eta_2(m), ..., \eta_n(m)\right), \left(m = 1, 2, ..., M\right),$$

and retain only those vectors $\eta(m)$ that satisfy the inequality

$$n_t(m) \geq k_t,\ \ t = 1, 2, ..., n-1. \tag{4.6}$$

Step 4

Denote by $\Psi_n(x)$ the conditional empirical cumulative distribution function of a random variable η_n that satisfies equation (4.6). The boundary b_n is determined as the quantile of the distribution $\Psi_n(x)$ corresponding to the conditional probability $\hat{P}_n$:

$$k_n = \Psi_n^{-1}\left(\hat{P}_n\right).$$

4.4.3 Convolution

Morokoff (2003) exploits the fact that from the second threshold onwards a truncated distribution is always convolved with a standard normal distribution.

This section begins with two theorems: the first theorem derives the convolution for the sum of two independent random variables and the second is the convolution theorem, which importantly uses Fourier transforms. The implementation and results are then discussed.

4.4.3.1 Theorem

If *X* and *Y* are independent continuous random variables, with density functions $f_X(x)$, $f_Y(y)$ respectively and $Z = X + Y$, then

$$f_z(z) = f_{X+Y}(z) = \int_{-\infty}^{\infty} f_Y(z-y) f_y(y) dy .$$

Proof: (Mood et al. (1974), p. 186).

4.4.3.2 The Convolution Theorem

Let $f_1(x)$ and $f_2(x)$ have Fourier transforms g_1 and g_2, respectively. Then the Fourier transform g of the convolution of $f_1(x)$ and $f_2(x)$, is the product $g(\omega) = g_1(\omega) g_2(\omega)$.

Proof: (Bracewell, 2000, p.117)

4.4.3.3 Convolution algorithm

The approach adopted is based on the convolution algorithm of Ruckdeschel et al. (2006)[2] contained in the *distr* module of the statistical software package R. Their implementation is for the *n*-fold convolution of continuous distributions.

The approach considered here is based on analytically determining the threshold from the conditional-default probability. This approach is an analytic approximation to the conditional probability approach of Iscoe & Krenin (unpublished a) and has the advantage of not requiring computationally expensive simulations.

The probability distribution of the asset return after say, n time-steps, is the convolution of the density at time $n-1$ with a standard normal distribution. Once the

[2] R is an open-source statistical program available at http://cran.r-project.org

distribution is known at time n, the threshold is computed such that the probability of being below the threshold is the conditional probability of survival over the period $n-1$ and n. The computational costs of direct convolution are quadratic. However, Morokoff (2003) propose the application of the convolution theorem and fast Fourier transforms (FFT) that reduces the computational cost. Cerny (2004) provides an introduction to the use of FFT to finance. Robertson (1992) and later Wang (1998) demonstrated the use of the FFT in actuarial science in determining the aggregate loss distributions. Robertson (1992, p. 64) demonstrates the use of discrete Fourier transforms (DFT) in determining convolutions. Due to the convolution theorem, convolutions can be determined with DFT, $\mathcal{F}$,

$$\mathcal{F}(U \otimes V) = \mathcal{F}(U) \times \mathcal{F}(V)$$
$$U \otimes V = \mathcal{F}^{-1}\left(\mathcal{F}(U) \times \mathcal{F}(V)\right).$$

Suppose we have the cumulative distribution function $P(n) = P(\tau \le t_n)$, $n = 1, 2, ..., N$ of the probability of default at discrete times $t_1, t_2, ..., t_N$, for asset i. The associated asset value of the log return process follows a Wiener process. A company i defaults during the period $[t_{d-1}, t_d)$ if the asset return $R_d^{(i)}$ at time T_d is less than some threshold k_d^i, while $R_d^{(i)} > k_d^i$ for all $i < d$, implies no previous default. The relationship to the cumulative default probabilities is,

$$1 - P\left(R_1^i > k_1^i, ..., R_d^i > k_d^i\right) = P(t_d).$$

Assuming that the distribution R_{d-1}^i, conditional on no defaults up to time t_{d-1}, is known, then the distribution of R_d^i, conditional on no defaults up to time t_{d-1}, is given by

$$R_d^i = R_d^{i-1} + \varphi_d$$

where φ_d is an independent increment with a normal distribution with zero mean and standard deviation $\sqrt{t_d - t_{d-1}}$.

Then the default threshold k_d^i may be obtained from the equation

$$P\left(R_d^i \le k_d^i \,\middle|\, R_1^i > k_1^i, \ldots, R_{d-1}^i > k_{d-1}^i\right) = \frac{P(t_d) - P(t_{d-1})}{1 - P(t_{d-1})}.$$

Once k_d^i has been determined, the distribution of R_d^i, conditional on no defaults up to t_d, can be determined by truncating the distribution of R_d^i conditionally on no defaults up to time $t_{d-1.}$ This procedure is repeated until all the thresholds are determined.

4.4.3.4 Default threshold determination algorithm using FFT

The actual implementation of an FFT algorithm is not discussed here as the FFT algorithm is 'perhaps, the most ubiquitous algorithm in use today' (Rockmore, 2000). The details of using the FFT to calculate the default thresholds are discussed below.

Step1

The first threshold as in section 4.4.2.3 is obtained analytically. Thus

$$k_1 = \Phi^{-1}\left(P(1)\right).$$

Step 2

An acceptable level of truncation error ε is chosen, and then the upper and lower bounds are found, such that

$$F(-\infty, l) = \frac{\varepsilon}{2} \text{ and } F(u, \infty) = \frac{\varepsilon}{2}.$$

Discretise the interval $[l, u]$ using some number of points, $M = 2^m$. Let

$$h = \frac{u - l}{2^m}.$$

Define the vector $\tilde{y}$, with values $y_j = l + jh, \ j = 0, 1, \ldots, 2^m$.

The distribution, conditional on there being no default in the previous period, is then used to determine the next threshold. This truncated distribution is convolved with a standard normal distribution. The algorithm given in section 4.4.3.3 is used to reduce the computational time of the convolution.

Step 3

Using the previous threshold and empirical distribution, truncate the distribution at the previous threshold. Truncation is defined by Loève (1977, p.245) as, truncate X at $c > 0$ (finite) when we replace X by $X^c = X$ or 0 according as $|X| < c$ or $|X| \geq c$, and X^c is X truncated at c. Denote the vector of the truncated cumulative distribution as $\Phi^{(i-1)}(\tilde{y})$. Then, obtain the value of the cumulative distribution function of a standard normal distribution, $\Phi(\tilde{y})$. Both these vectors have $M+1$ elements.

Step 4

Obtain the probability density function of $\Phi^{(i-1)}(\tilde{y})$ and $\Phi(\tilde{y})$ from the above vectors. For example the probability distribution function of the normal distribution is:

$$\phi(y_j) = \Phi(y_j) - \Phi(y_{j-1}),\ j = 0,1,\ldots,2^m .$$

This discretisation is called ‘forward difference’ (Dutang et al, 2008).

The size of the vectors is now M. Let these vectors by p_1 and p_2 respectively and let $\tilde{\mathcal{F}}$ be the FFT implementation of $\mathcal{F}$. The probability density function, f, of the convolution is given by

$$f = \tilde{\mathcal{F}}^{-1}\left(\tilde{\mathcal{F}}(p_1)\tilde{\mathcal{F}}(p_2)\right)$$

The cumulative sum of this vector is the cumulative distribution function.

Step 5

Additional accuracy is reported by applying a continuity correction (Kohl et al, unpublished).

The values are linearly interpolated on $\{2l+0.5h, 2l+1.5h, \ldots, 2u-0.5h, 2u\}$ to get a continuous approximation of the distribution function $F = F_1 \otimes F_2$.

Step 6

Once F is known, the threshold k_i is determined by the inverse cumulative distribution function or quantile function. This threshold is used to truncate the empirical cumulative distribution function F. This conditional cumulative distribution function is used in Step 3 to determine the next threshold. This process is repeated until the final threshold is calculated.

4.4.4 Numerical comparison of the methods

In this section numerical experiments are conducted. The objective is to identify the computationally most efficient method subject to an acceptable level of accuracy. The following measures of accuracy are used:

Definition

Let $\hat{x}$ be an approximation to a real number x. The absolute error

$$E_{abs}\left(\hat{x}\right) = \left|x - \hat{x}\right|,$$

and its relative error

$$E_{rel}\left(\hat{x}\right) = \frac{\left|x - \hat{x}\right|}{\left|x\right|}.$$

A simple hypothetical example is first used to test the relative accuracy of all the methods outlined above using a direct convolution method.

The cumulative default probabilities of a hypothetical entity are listed in Table 4.1. The conditional default probabilities, $\hat{P}_n$, are obtained from the cumulative default probabilities, $P(n) = P(\tau \le t_n)$, at times $t_n, n = 1, \ldots 5,$ and τ is the time of default defined previously, by application of:

$$\hat{P}_n = \frac{P(n) - P(n-1)}{1 - P(n-1)}.$$

Table 4.1 Conditional and cumulative default probabilities

Term (Years)	1	2	3	4	5
Cumulative Probability	5%	10%	15%	20%	25%
Conditional Probability	5.00%	5.26%	5.56%	5.88%	6.25%

The default thresholds are given in Table 4.2 based on the data given in Table 4.1. This was determined using a direct convolution. As the direct method is deemed most accurate, the results of the previously described methods of Hull & White (2001), Iscoe & Krenin (unpublished a) and Morokoff (2003) are compared to it.

Table 4.2 Default thresholds obtained via direct convolution

Term (Years)	Default threshold
1	-1.64485
2	-2.03202
3	-2.17211
4	2.181324
5	-2.10199

The direct convolution is impractical due to the extraordinary amount of computational time required. The results of this method were obtained using the *quadgk* function[3] in Matlab, and it took approximately 34 hours to calculate. In a practical example of a 5-year CDO contract, which typically has quarterly payment dates, there are 20 thresholds to compute for each entity in the portfolio. The use of the direct method is therefore unfeasible.

4.4.4.1 Relative error determination

For the approach of Hull & White (2001), the number of discrete points is the main parameter determining accuracy, as is the case for Morokoff (2003). The main parameter

[3] *quadgk* is Matlab's implementation of the adaptive Gauss-Kronrod quadrature.

for Iscoe & Krenin (unpublished a) is the number of simulations. In this section these parameters are successively increased until the maximum accuracy is achieved for each approach.

Hull & White (2001) suggested using 'several hundred' discrete points. For verification, the parameter M was increased by 100 and the relative errors were noted. The results are presented in Figure 4.2 below.

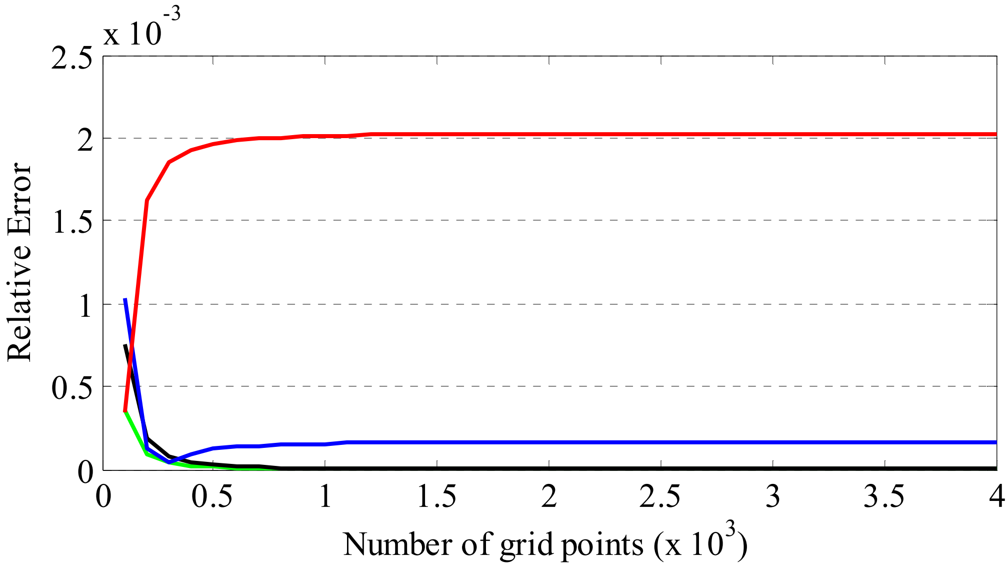

Figure 4.2: Relative error of the Hull & White default thresholds

The relative error appears constant when the number of grid point is greater than 700. However, the results indicate an increase in relative error as the number of thresholds is increased. The behaviour of the 5th threshold could be attributed to round-off error. This assertion could be tested by doing all calculations using a multi-precision environment.[4]

For the approach of Iscoe & Kreinin (unpublished a) (illustrated in Figure 4.3) a standard Monte Carlo approach was implemented. A very large number of simulations (of 5×10^6) is required to achieve relative errors of less than 2×10^{-3}.

[4] An extended precision toolbox for Matlab is available at http://www.mathworks.com/matlabcentral/fileexchange/loadFile.do?objectId=6446. This allows for arbitrary precision.

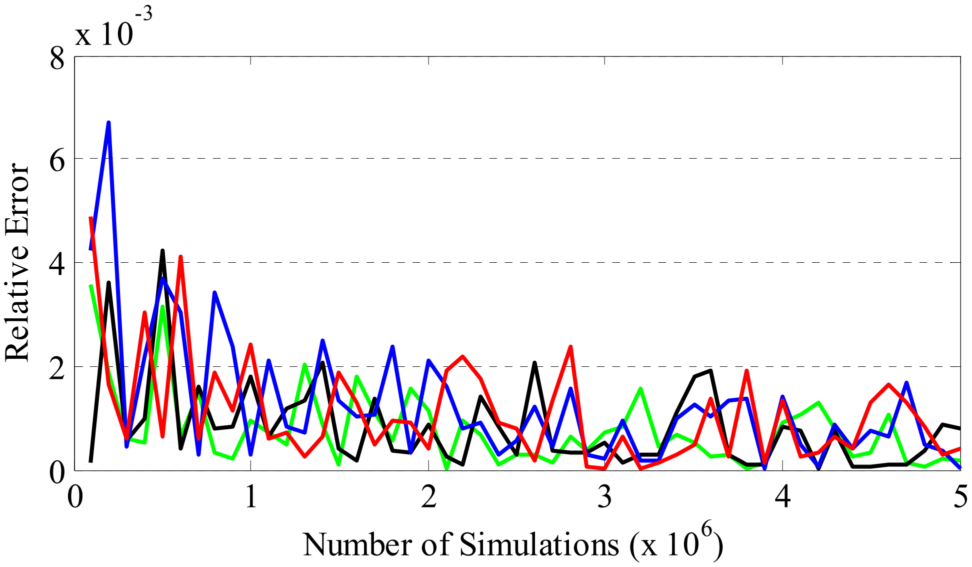

Figure 4.3: Iscoe & Kreinin, plain Monte Carlo approach

A simple way of improving Monte Carlo estimates is by using antithetic variates (Ross, 2006). This method saves computational time by effectively doubling the number of random numbers generated, provided the variates are negatively correlated.

Figure 4.4 below, demonstrates the decrease in the relative error brought about by the usage of antithetic variates. The relative error using antithetic variates with 3×10^6 simulations is similar to that obtained using the vanilla Monte Carlo approach with 5×10^6 simulations. Thus using antithetic variates the number of simulations chosen is 3×10^6.

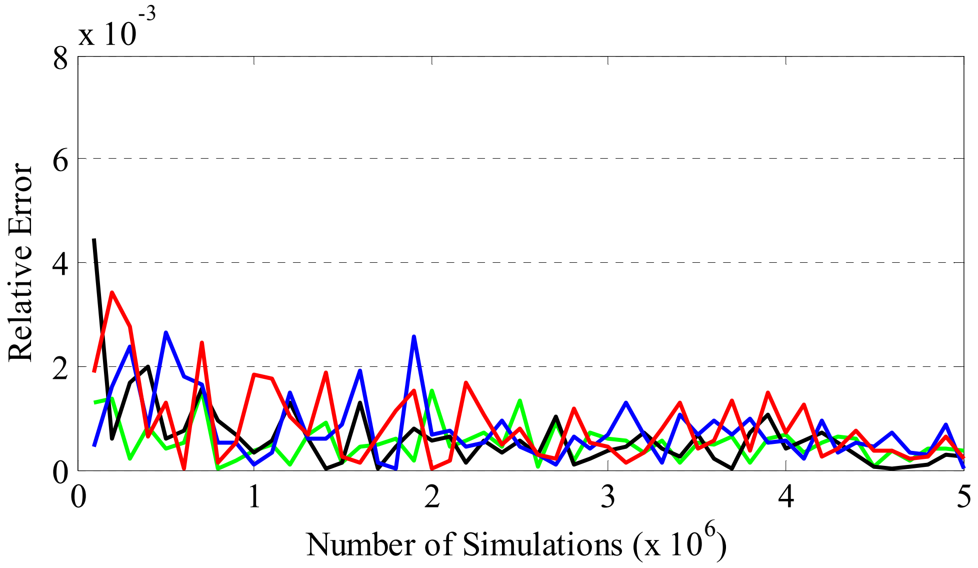

Figure 4.4: Iscoe & Kreinin, Monte Carlo using antithetic variates

The relative error of the approach by Morokoff (2003), illustrated in Figure 4.5, is vastly superior to that of the competing methods. A parameter value of 2^{16} is adequate for relative error to be less than 1.5×10^{-7}. As with the other methods the relative error increases as the threshold number increases.

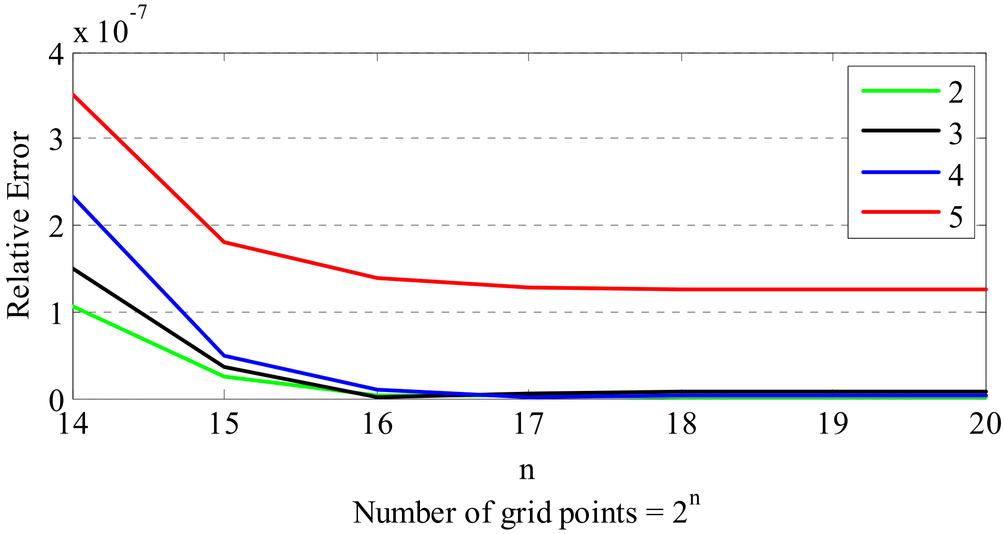

Figure 4.5: Morokoff approach using FFT

The results for the relative errors obtained when compared to that obtained by direct convolution are summarised in Table 4.3. The method of Morokoff (2003) produces the smallest relative error.

Table 4.3: Relative errors of the 3 methods relative to the direct method

Hull & White	Iscoe & Kreinin	Morokoff
7.21E-06	6.04E-04	2.98E-09
1.32E-05	3.68E-04	2.11E-10
1.44E-04	6.76E-04	9.57E-09
2.00E-03	4.64E-04	1.39E-07

4.4.4.2 Computational efficiency

The computational time depends directly on the number of simulations or the grid points used. Once the accuracy has been established, the focus then is on computational efficiency. The methods described above together with the parameters chosen are compared with respect to computational efficiency. This is accomplished by measuring the total computational time. The results are in Table 4.4 below.

Table 4.4: Comparison of the computational times in seconds

Hull & White	Iscoe & Kreinin	Morokoff
1.64	2.11	0.87

4.4.4.3 Summary

The results obtained above could be summarised as follows. The simulation method of Iscoe & Kreinin (unpublished a) is slower but more accurate than that of Hull & White (2001). Morokoff (2003) is both more accurate and faster than the techniques that preceded it. This supports the statement by Arvanitis & Gregory (2001, p. 162):

> In equation (4.2) the calculation of the thresholds can be done numerically via the same Monte Carlo simulation as is used for the pricing. Alternatively, an analytical approach can be used, such as that described by Hull and White (2001). We have found that the numerical calibration of the thresholds is slower but gives superior accuracy.

The issue of pricing of CDO tranches is to be dealt with in the next section. The technique of Morokoff (2003) appeared after the work of Arvanitis & Gregory (2001) and given its superior performance in terms of accuracy and computational efficiency, this technique is to be used for pricing of synthetic CDO tranches.

4.5 CDO valuation

By using the default thresholds we are now able to value a synthetic CDO. In this section a description of how a CDO is evaluated using the multi-step approach is provided. In addition the three methods described earlier to determine the default thresholds are used. The aim is to evaluate to what extent the accuracy of the default thresholds affects CDO tranche pricing.

4.5.1 Loss distribution and the pricing of CDOs

The description in this section is based on that of Cherubini et al. (2004). Consider n reference obligors with nominal amount A_i and a recovery rate R_i with $i = 1,\ldots n.$ $L_i = \left(1 - R_i\right)A_i$ denotes the loss given default or the net loss for the ith credit. Let τ_i be the default time of the ith name and $N_i\left(t\right) = \mathbf{1}_{\{\tau_i < t\}}$ be the counting process which jumps from 0 to 1 at the default time of name i. Then $L\left(t\right)$, which is a pure jump process, denotes the cumulative loss on the collateral portfolio at time t :

$$L\left(t\right) = \sum_{i=1}^{n} L_i N_i\left(t\right).$$

Let us consider a tranche of a CDO, where the default leg pays all the losses that occur on the collateral portfolio above a threshold C and below a threshold D, where $0 \le C \le D \le \sum_{i=1}^{n} A_i$.

When $C = 0$ the tranche in consideration is the equity tranche; if $C > 0$ and $D < \sum_{i=1}^{n} A_i$ we are considering the mezzanine tranche. When $D = \sum_{i=1}^{n} A_i$ we consider the senior tranche.

Let $M(t)$ be the cumulative loss on a given tranche, hence,

$$\begin{aligned} M(t) &= \begin{cases} 0 & \textbf{if } L(t) \le C \\ L(t) - C & \textbf{if } C < L(t) \le D \\ D - C & \textbf{if } L(t) > D \end{cases} \\ &= \min\left(\max\left(D - L(t), 0\right), D - C\right) \end{aligned}$$

or equivalently:

$$M(t) = \left(L(t) - C\right)\mathbb{1}_{(C,D]}L(t) + \left(D - C\right)\mathbb{1}_{\left(D,\sum_{i=1}^{n} A_i\right]}L(t).$$

Given that $L(t)$ is a jump process, it follows that $M(t)$ is a jump process as well. There is a default payment on every jump of $M(t)$. A Riemann-Stieltjes integral could be defined with respect to $M(t)$, as it is an increasing process. As $M(t)$ is constant between jump times, the Riemann-Stieltjes integral with respect to $M(t)$ is a discrete sum with respect to every jump time.

Let $B(t)$ be the discount factor for the maturity t, and let T denote the maturity of the CDO. The value of the default leg of the given tranche is

$$\mathbb{E}\left[\int_0^T B(t)\,dM(t)\right],$$

The value of the premium leg t time 0, assuming that the premium s is paid continuously is

$$s\mathbb{E}\left[\int_0^T B(t)\,g\left(L(t)\right)dt\right],$$

where $g\left(L(t)\right) = \min\left(\max\left(D - L(t), 0\right), D - C\right)$. The fair value of the premium s of the tranche is given by

$$s = \frac{\mathbb{E}\left[\int_0^T B(t)\,dM(t)\right]}{\mathbb{E}\left[\int_0^T B(t)\,g(L(t))\,dt\right]}.$$

Assuming that the portfolio is evaluated at m discrete premium dates, $\mathrm{t}_0 = 0, t_1, \ldots, t_m = T$ then the above expression becomes

$$s = \frac{\sum_{i=1}^{m}\left(\mathbb{E}\left[g\left(L(t_i)\right)\right] - \mathbb{E}\left[g\left(L(t_{i-1})\right)\right]\right)B(t_i)}{\sum_{i=1}^{m}\left(t_i - t_{i-1}\right)\left(S - \mathbb{E}\left[g\left(L(t_i)\right)\right]\right)B(t_i)},$$

which is similar to equation (1.1)

4.5.2 Description of the multi-step model

The default thresholds for all the assets in the portfolio are calculated. The pool of assets is simulated over the first period. The defaults, if any, are tabulated, then the remaining non-defaulting assets are simulated over the next period and checked if any has defaulted. This process continues until the maturity date of the contract.

4.5.3 Assumptions

The multi-step model described thus far assumes that:

- default can happen at any set of discrete dates;
- the recovery rate is constant;
- the default process is independent of the interest-rate;
- the interest-rate is deterministic;
- the hazard rates are constant over the term of the contract;
- default correlation between companies is constant throughout the term of the deal.

4.5.4 Algorithm

Assume that we have a portfolio of n assets, and a CDO contract that has premium payment dates $t_j,\ j = 1,\dots,S$ The algorithm to determine the fair tranche spreads is illustrated below.

Step 1

Generate the correlated random Brownian motion for each of the n companies in the portfolio at each of the S time-steps. These numbers are stored in a matrix A that has dimension $n \times S$. The path of the Brownian motion till time t_S is given by the sum across the columns of A. Define a matrix D that has dimensions $n \times 2$. Set all the elements of D to zero. The first column of D represents the state of the company, the value one represents default and zero non-default. The second column is left for the time of default.

Step 2

For each company $i, i = 1,\dots,n$ in the portfolio, and at each time-step $t_j,\ j = 1,\dots,S$, evaluate $D\left(i, j\right) = 0$. If this is false then the company has already defaulted and the next company is considered. If this is true then check if $A\left(i, j\right) < k\left(i, j\right)$ is true. If false, then consider there is no default. If true then update the matrix D by changing its default state and storing the time of default, $D\left(i,1\right) = 1$ and $D\left(i,2\right) = t_j$.

At the end of checking each company at each time step, the default times and the default amounts are known.

Step 3

The matrix D is then used with the equation (1.1) to determine the present value of the premium and default legs.

Step 4

The present values are kept in a running total.

Steps 5

Steps 1 through 4 are repeated M times, where M is the number of simulations.

For the example that follows, it is assumed that the default correlation between each company in the portfolio is 50% and the recovery rate is constant at 30%. The portfolio is split into four tranches covering losses: 0–25, 25–75, 75–150 and 150–400. The other parameters for the model are a term of five years with quarterly payments. The portfolio is presented in Table 4.5.

4.5.5 Default leg

For each simulation, the default time and the defaulted asset amounts are collected, in $2 \times n$ matrix, where n is the number of defaults. The discounted value of the default leg of the CDO is determined via this matrix of times and the associated defaulted amount (net of recoveries).

4.5.6 Premium leg

The premium is paid on the remaining size of outstanding portfolio. As defaults increase the size of the premium reduces proportionately.

Table 4.5: The data for credit default swap portfolio

Company	Notional	Basis Points	Company	Notional	Basis Points
1	15	40	26	5	130
2	15	50	27	10	220
3	15	50	28	5	300
4	15	50	29	5	160
5	10	60	30	10	100
6	10	60	31	5	80
7	10	80	32	10	130
8	10	70	33	5	120
9	10	90	34	10	190
10	10	100	35	5	120
11	10	90	36	5	125
12	10	80	37	5	80
13	5	60	38	5	250
14	5	50	39	5	100
15	5	80	40	10	250
16	5	60	41	5	165
17	10	100	42	5	250
18	10	60	43	10	190
19	5	80	44	10	150
20	5	60	45	10	180
21	5	70	46	5	155
22	10	150	47	10	190
23	5	110	48	5	250
24	10	200	49	5	375
25	10	170	50	5	550

4.5.7 Results from the methods

In this section the expected loss of four tranches of a heterogeneous portfolio (as defined earlier) of a CDS is evaluated. Table 4.6 illustrates the fair spreads obtained by simulation. The simulation consisted of 100 000 simulation runs. Although the Hull & White approach used to calculate the default boundaries was found to be inaccurate, this did not have a significant impact on the results apart from a small increase in the equity tranche spread relative to the other approaches.

Table 4.6: Fair tranche spread comparison

Tranche	Hull & White	Iscoe & Kreinin	Morokoff
0–25	1382.18	1362.90	1369.34
25–75	364.86	359.45	357.22
75–150	82.51	82.90	80.86
150–400	3.79	3.60	3.69

Table 4.7 below produces the relative error of the fair tranche spreads of the Hull & White (2001) and Iscoe & Kreinin (unpublished a) relative to that of Morokoff (2003), which was found to be the most accurate method.

Table 4.7 : The fair tranche spreads relative to that obtained using Morokoff (2003)

Tranche	Hull & White	Iscoe & Kreinin
0–25	0.94%	0.47%
25–75	2.14%	0.62%
75–150	2.04%	0.74%
150–400	2.71%	1.36%

4.6 Conclusion

In this chapter the multi-step model for determining the credit loss distribution and using it to value a four-tranche CDO was explored. In addition three schemes were discussed for determining the default boundary. A simple contrived example was used to demonstrate that the direct convolution method for determining the default thresholds is impractical as it is computationally expensive. The method of Iscoe & Krenin (unpublished a) was enhanced by using the simple technique of antithetic variates.

The approximation of the convolution by FFT was identified as the fastest and most accurate method of determining the default thresholds.

It was found that the accuracy of the three methods did impact the results of the pricing of CDO tranches. Using the result of Morokoff (2003) as the most accurate, the Hull & White (2001) method produced errors that increased from approximately 1% for the equity tranche to 3% for the senior tranche. The range for the method of Iscoe & Kreinin (unpublished a) was approximately 0.5–1.4% (Table 4.7).

In the next chapter the default time model of Li (2000) is introduced and compared in terms of accuracy and computational efficiency with the multi-step approach.

Chapter 5

Monte Carlo methods Part II: Default-time approach

A multi-step model was introduced in the previous chapter, which allowed for the time of default to be determined implicitly: for each obligor in the portfolio, the first instance of a simulated Brownian-motion path crossing a certain boundary or threshold was taken to be the time of default. Thus two calculations are needed: one to determine the thresholds and another to simulate Brownian-motion paths to determine the time of default. In the default-time model of Li (2000) the default times of the entities in a portfolio were determined explicitly. Hence this model is called ‘single-step’. This model is compared with that of the multi-step model with respect to accuracy and computational efficiency. It is found that although the default-time approach is much faster, there are differences in the results. An attempt is made to explain the results.

5.1 The copula function

A copula is used, as a general way of formulating a multivariate distribution, in such a way that various types of dependence may be represented (Nelson, 1999).

5.1.1 Skar’s Theorem

A theorem attributed to Sklar (1959) is ‘central to the theory of copulas, and is the foundation of many, if not most, of the applications of that theory to statistics’ (Nelson, 1999, p. 14).

Theorem 5.1 Sklar’s Theorem

Let F be a multivariate m-dimensional distribution function with marginals $F_1,...,F_m$. Then there exists a copula function C such that

$$F\left(x_1,...,x_m\right)=C\left(F_1\left(x_1\right),...,F_m\left(x_m\right)\right) \qquad \left(x_1,...,x_m \in \mathbb{R}\right).$$

Conversely, for any copula C and marginal distribution functions $F_1,...,F_m$, the function

$$F(x_1,...,x_m) = C(F_1(x_1),....,F(x_m)) \qquad (x_1,...,x_m \in \mathbb{R})$$

defines a multivariate distribution function with marginals $F_1,....,F_m$

(Blum, 2003, p.104).

Proof: (Nelson, 1999, p.15)

Of particular interest is the Gaussian copula. The multivariate copula function

$$C(u_1,u_2,...,u_m) = \Phi_m\left(\Phi^{-1}(u_1),\Phi^{-1}(u_2),...,\Phi^{-1}(u_m),\Sigma\right)$$

where Φ_m is the m-dimensional normal distribution function with correlation coefficient matrix Σ, Φ is the univariate normal distribution function and Φ^{-1} is the inverse of the univariate normal distribution function.

5.1.2 General simulation algorithm

This section provides details of an algorithm used to generate defaults using the multivariate normal copula. It is based on Li (2000).

We assume that we have a portfolio of n credits, and that each credit has a hazard-rate function for survival time T_i. The distribution function of T_i is $F_i(t)$. Using a copula function C we also obtain the joint distribution of the survival times as

$$F(t_1,t_2,...,t_n) = C(F_1(t_1),F_2(t_2),...,F_n(t_n)).$$

If the normal copula function is used then we have

$$F(t_1,...,t_n) = \Phi_n\left(\Phi^{-1}(F_1(t_1)),...,\Phi^{-1}(F_n(t_n))\right),$$

where Φ_n is the n-dimensional cumulative distribution function with correlation coefficient matrix Σ.

To simulate the correlated survival times we introduce another series of random variables $Y_1,Y_2,...,Y_n$ such that

$$Y_1 = \Phi^{-1}(F_1(T_1)), Y_2 = \Phi^{-1}(F_2(T_2)),...,Y_n = \Phi^{-1}(F_n(T_n)).$$

There is a one-to-one mapping between $T_1, T_2, ...T_n$ and $Y_1, Y_2, ...Y_n$. Simulating $\{T_i | i = 1, 2, ..., n\}$ is equivalent to simulating $\{Y_i | i = 1, 2, ..., n\}$. The simulation algorithm is as follows:

Step 1

Simulate $Y_1, Y_2, ...Y_n$ from an n-dimensional normal distribution with correlation coefficient matrix Σ. As was the case in the multi-step model, correlated random variables need to be generated.

Step2

Obtain $T_1, T_2, ..., T_n$ using $T_i = F_i^{-1}\left(\Phi\left(Y_i\right)\right)$, $i = 1, ..n$. This step requires the inverse cumulative distribution function (cdf) of the default distribution. In chapter 2 this was given as

$$F\left(t\right) = 1 - \exp\left(-\int_0^t h du\right)$$

where h is the constant hazard rate. The inverse cdf of the above (exponential distribution) is

$$F^{-1}\left(\Phi\left(Y\right)\right) = \frac{-\left(1 - \Phi\left(Y\right)\right)}{h},$$

noting that $0 \leq \Phi\left(Y\right) \leq 1$.

With each simulation the default times for all the entities in the portfolio are known. As in the previous chapter, the time of default and the amount at default is all the information needed to determine the loss distribution that is used to evaluate CDO tranches.

5.2 Pricing of synthetic CDO tranches

The portfolio used in chapter 4 to demonstrate the multi-step pricing model is used. The two methods are applied to the portfolio. A computationally efficient method to determine the default thresholds was identified as the convolution technique of Morokoff (2003). The number of grid points used was 2^{12}. The 95% nonparametric confidence interval for the credit spreads was found by using the 2.5th and the 97.5th percentile of the simulated results. The simulation run (each consisting of 100 000) was repeated 100 times with different seeds and the midpoint of the confidence interval was then taken as the price for that tranche rounded to the nearest basis point (De Prisco et al, 2005). The tranche spreads of the above simulation are presented in Table 5.1, where the confidence interval is denoted by CI and the Monte Carlo simulation result is denoted by MC. The differences in the results are considered in the next section.

Table 5.1: Comparison of the tranche spreads, in basis points, between the multi-step and default-time approaches

Tranche attachment points		Multi-Step (MS)		Default-time (DT)		Relative (MS-DT)% error
Lower	Upper	CI	MC	CI	MC	
0	25	[1348,1372]	1360	[1280,1299]	1289	5.2
25	75	[353,362]	358	[358,367]	362	-1.1
75	150	[79,83]	81	[88,93]	91	-12.3
150	400	[4,4]	4	[5,5]	5	-25.0

In assessing the computational efficiency of each approach the simulation time was recorded for each tranche for the 100 simulation runs. The average time of these 400 observations is reported in Figure 5.2 for the respective approaches.

Table 5.2: The average computational time in seconds

Multi-Step	Default-time
41.3	15.5

The default-time approach is computationally more efficient than the multi-step approach.[5]

5.2.1 Comparison of the default-time and multi-step approaches

In order to explain the differences of the results, a numerical exercise was conducted. For this exercise 10^6 simulations were conducted and the probability density function (pdf) was constructed for both methods. The two methods yield slightly different probability distributions. The results are shown in Figure 5.1.

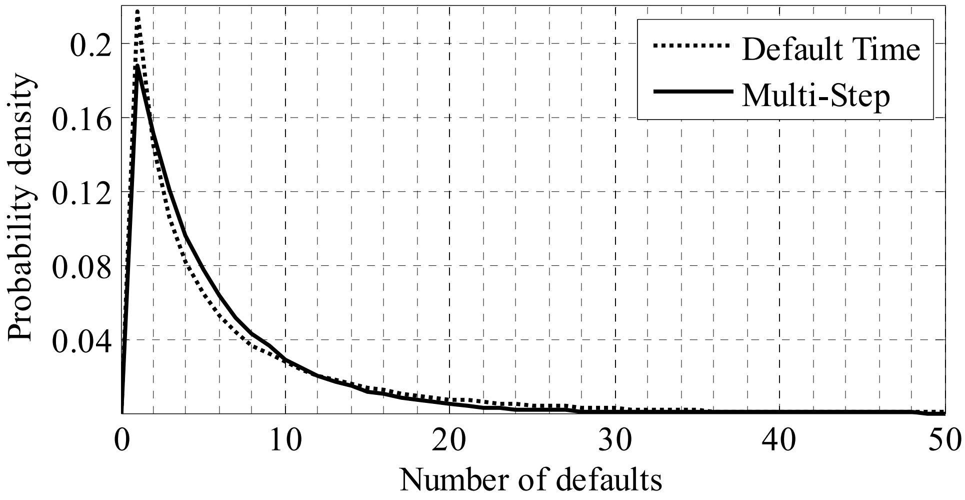

Figure 5.1 Comparison of the default-time and multi-step approach for generating defaults

The graph demonstrates that the default-time approach produced a higher probability of a single default than the multi-step method. However, the probability of two or more defaults is less than that of the multi-step approach. In the upper tail region of the pdf, which is applicable to the upper or senior tranches of a CDO, the probability of default is higher for the default-time method. This appears to explain the results for the tranches in Table 5.1. The default-time method produces lower values for the equity tranche and higher values for other tranches. The graph in Figure 5.2 from Morokoff (2003) compares the cumulative distribution function of the two approaches using a high-yield portfolio (i.e. high default probability). The results for the portfolio used above are similar to that in Figure 5.2.

[5] The computer used has a 1.66GHz dual core processor and 1 Gigabyte of RAM.

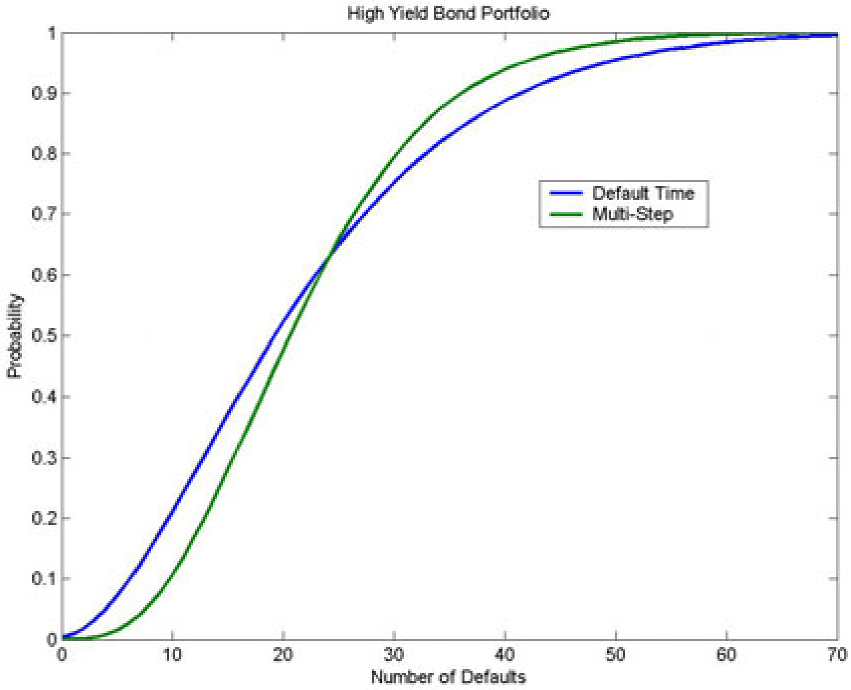

Figure 5.2 The cdf of the default-time and multi-step approaches for generating correlated defaults (Source: Morokoff (2003))

5.3 Conclusion

In this chapter a four-tranche CDO was evaluated using the approach of Li (2000) and the multi-step approach described in the previous chapter. The differences between these approaches were significant in the tail regions, whilst the default-time approach was significantly faster.

The Monte Carlo simulation method is the most general of all approaches and is applicable to all types of CDO structures, however it suffers from the disadvantage of being computationally expensive. As the valuation of CDO tranches amounts to the determination of the loss distribution, further research into efficient analytic methods to determine the loss distribution has been undertaken. These techniques that are used in the insurance industry to determine the aggregate loss of claims are to be explored in the next chapter.

Chapter 6

Semi-analytic methods

This chapter introduces and compares a variety of analytic approximations that are used to determine the credit loss distribution. These approximations require numerical solutions as opposed to the Monte Carlo simulation; hence they are termed 'semi-analytic'. The methods considered here in the credit-derivatives context have traditionally been used to calculate the aggregate loss distribution for insurance claims.

The computational efficiency and accuracy (relative to the results of the previous chapter) are examined. The methods used here are the numerical inversion of characteristic functions, exact recursion and the compound Poisson approximation.

6.1 Introduction

The motivation for the single-factor approach is to derive a solution without resorting to simulation, which is computationally expensive. The contribution to the literature of this chapter is that the valuation methods are collated, described and compared. In addition the presentation differs from that of the original authors in that the description here is given from an actuarial perspective and is influenced by the presentation of Klugman et al. (2004, chapter 6).

This chapter is organised as follows. The next section describes the single-factor model of Vasicek (unpublished a, 2002) and the conditional-independence framework. Section 6.3 introduces the three semi-analytic methods: the numerical inversion of characteristic functions, exact recursion and the compound Poisson approximation. Each of these methods is implemented and the values of the CDO tranches and computational time recorded. These results are compared with those obtained in the previous chapter. Section 6.4 provides a summary of the results and the chapter concludes with section 6.5.

6.2 The single-factor model

Let N be the number of assets in a portfolio with the *k*th asset having a default probability of p_k for a given time horizon *T*. This default probability is for the same horizon as that of the desired portfolio loss distribution. Each asset has an associated normalised credit-risk indicator, V_k, for the horizon T. The lower V_k is, the higher the credit risk of asset k. The asset k defaults if V_k falls below the default threshold α_k. As V_k is normally distributed, $\alpha_k = \Phi^{-1}\left(P\left(V_k < \alpha_k\right)\right) = \Phi^{-1}\left(p_k\right)$.

In the single-factor model the credit risk of each asset can be decomposed into systemic and an idiosyncratic risk. By 'systemic', as the word implies, this risk is attributed to macro-economic factors, such as interest rates or a financial index. The idiosyncratic risk is also called 'specific risk'. This risk is peculiar to that asset. The systemic risk factor V and the idiosyncratic risk factor ε_k are assumed to be independent normally distributed random variables with zero mean and unit variance (Finger, unpublished).

The central assumption of the single-factor model is that the credit risk is assumed to be:

$$V_k = w_k V + \theta_k \varepsilon_k$$

where w_k is the correlation between V and V_k and θ_k is the weight of the idiosyncratic risk for the kth asset. Given that V_k, V and ε_k are all normally distributed, $\theta_k = \sqrt{1-w_k^2}$. This is obtained by direct application of the following result that the distribution of a linear function of n independent random variables $x_1 \dots x_n$ where x_j is distributed normally with zero mean and variance σ_j^2 is normally distributed. Let the linear function be:

$$z = a_1 x_1 + \dots + a_n x_n,$$

where a_j are constants, then $z \sim N\left(0, \sum a_j^2 \sigma_j^2\right)$ (e.g. Stuart & Ord, 1994, p.378).

6.2.1 The conditional-independence framework

We assume that the change in the normalised asset value can be expressed by:

$$V_k = w_k V + \sqrt{1-w_k}\,\varepsilon_k .$$

From the above $P(V_k < \alpha_{\mathrm{k}}) = F_k(t)$, where $F_k(t)$ is the probability of default occurring before time t for asset k, and since V_k has a standard normal probability distribution, then $\alpha_k = \Phi^{-1}(F_k(t))$. For a given state of the economy V, the kth asset will default if $V_k < \alpha_k$, i.e.

$$\varepsilon_k < \frac{(\alpha_k - w_k)}{\sqrt{1-w_k^2}}.$$

Given that ε_k is normally distributed with zero mean and unit variance, the default probability of the kth debtor, conditional to V, is given by

$$p_t^{k|V} = \Phi\left(\frac{(\alpha_k - w_k)}{\sqrt{1-w_k^2}}\right).$$

6.2.2 The conditional independence applied to CDO valuation

Once we have obtained the conditional default probability, we have three broad approaches to the determination of the unconditional probability-of-loss distribution that have found widespread use amongst market practitioners. In all these approaches, the entire loss distribution is calculated. The problem of computing the value of synthetic CDO tranches is equivalent to determining the portfolio loss distribution. In this chapter the problem is solved using semi-analytic techniques. This is meant to be without the use of Monte Carlo, but with the use of numerical methods. The problem is stated mathematically as follows, where the probability of loss x of the portfolio at time t is given by

$$\int P\left(L(t)\middle|V = x\right)dV,$$

where $L(t) = X_1 + X_2 + \ldots + X_n$, n is the number of loans in the portfolio and $X_k, k = 1,\ldots,n$ are the notional amounts of the loans in the portfolio. Once the loss distribution is known at any time *t,* the fair tranche spread s_T for maturity T is given by equation (1.1).

6.3 Computation of the unconditional loss distribution

This section defines the individual and collective risk models. These models, traditionally used in risk theory, form an appropriate building block for the development of the semi-analytic formula.

Definition 6.1 Collective risk model

Let S denote the aggregate losses as a sum of a random number N of individual payment amounts $\left(X_1, X_2, ..., X_N\right)$. Then the collective risk model has the following representation:

$$S = X_1 + X_2 + ... + X_N, \quad N = 1, 2, ...,$$

where $S = 0$ when $N = 0$, and $X_j, \ j = 1, 2, ..., N$, are independent identically distributed random variables, unless otherwise specified. The independence assumptions are:

1. For a given value of $N = n$, the random variables $X_1, X_2, ..., X_n$ are i.i.d. random variables.
2. For a given value of $N = n$, the common distribution of the random variables $X_1, X_2, ..., X_n$ does not depend on n.
3. The distribution of N does not depend in any way on the values of $X_1, X_2, ..., X_n$

Definition 6.2 Individual risk model

The individual risk model represents the aggregate loss as a sum $S = X_1 + X_2 + ... + X_n$ of a fixed number, n, of insurance or financial derivative contracts. The loss amounts for the n contracts are $\left(X_1, X_2, ... X_n\right)$ where the $X_j, \ j = 1, 2, ..., n$ are assumed to be independent but not assumed to be identically distributed. The distribution of the $X_j, \ j = 1, 2, ..., n$ usually has a probability mass at zero, corresponding to the probability of no loss or payment.

(Klugman et al, 2004, p. 134-135).

As presented above, once the factor is fixed, the determination of the conditional loss distribution is similar to the determination of the aggregate loss distribution in the individual risk model. In this section the following methods are illustrated: inversion of the characteristic function, exact recursion, and compound Poisson approximation to the individual risk model. Conditional on a particular scenario the numerical inversion of the characteristic function and the exact recursion methods are essentially the valuation of the loss distribution in an individual risk model. The (unconditional) aggregate loss distribution is then evaluated at all possible scenarios. This requires numerical integration. The performance of these techniques is compared using the same portfolio of names given in chapter 4, with respect to accuracy and computational efficiency.

6.3.1 Numerical inversion

This approach to pricing CDO tranches was proposed by Laurent & Gregory (2003). It is based on the single-factor model and a semi-analytic formula is derived that avoids computationally expensive Monte Carlo simulations. An overview of the properties of characteristic functions is given, followed by a derivation of the characteristic function under the conditional-independence framework. A discussion of the theory and numerical implementation issues then follows. Klugman et al. (2004, p.191) describes the approach followed here without further elaboration:

> Through the use of transforms, both the FFT and inversion methods are able to handle convolutions efficiently. For example, suppose a reinsurance agreement was to cover the aggregate losses of three groups, each with unique frequency and severity distributions. If $S_i,\ i = 1,2,3$ are the aggregate losses for each group, the characteristic function for the total aggregate losses $S = S_1 + S_2 + S_3$ is $\varphi_S(z) = \varphi_{S_1}(z)\ \varphi_{S_2}(z)\ \varphi_{S_3}(z)$ and so the only extra work is the inversion step.

6.3.1.1 Properties of characteristic functions

Selected properties are introduced pertaining to the conditions that are necessary for the existence of characteristic functions. The inversion formula is also described.

Definition 6.3 Characteristic function

The characteristic function of a random variable X is

$$\varphi_X\left(u\right) = E\left(e^{iuX}\right) = E\left(\cos uX + i\sin uX\right),$$

where $i = \sqrt{-1}$ (Chung, 2001, p.150).

Definition 6.4

For any continuous function $f\left(x\right)$, the Fourier transform is the mapping

$$\tilde{f}\left(u\right) = \int_{-\infty}^{\infty} f\left(x\right)e^{iux}dx$$

(e.g. Panjer, 2006, p. 184).

The original function can be recovered from its inverse Fourier transform as

$$f\left(x\right) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \tilde{f}\left(u\right)e^{-iux}du.$$

When $f\left(z\right)$ is a probability density function, $\tilde{f}\left(z\right)$ is its characteristic function. If $f\left(x\right)$ is a probability mass function, with discrete probabilities at $x_j,\ j = 1,2,...$ then its Fourier transform, $\tilde{f}\left(z\right)$ is defined as

$$\tilde{f}\left(u\right) = \sum_{j=1}^{\infty} f\left(x_j\right) e^{iux_j}$$

(e.g. Stuart & Ord, 1994, p. 128).

The theorem below provides a one-to-one correspondence between characteristic functions and distribution functions.

Theorem 6.1 The uniqueness theorem

Two distribution functions $F_1\left(x\right)$ and $F_2\left(x\right)$ are identical if, and only if, their characteristic functions $\phi_1\left(u\right)$ and $\phi_2\left(u\right)$ are identical.

Proof: Lukacs (1970, p. 28)

Theorem 6.2 The inversion theorem

Let $f(t)$ be the characteristic function of the distribution function $F(x)$. Then

$$F(a+h)-F(a)=\frac{1}{2\pi}\int_{-\infty}^{\infty}\frac{1-e^{-ith}}{it}e^{-ita}f(t)\,dt,$$

Provided that a and $a+h$ $(with\ h>0)$ are continuous functions. (Lukacs, 1970, p.31)

Theorem 6.3 Inversion formula for density functions

If a characteristic function $\phi(t)$ is absolutely integrable over $(-\infty,+\infty)$ then the corresponding distribution function $F(x)$ is continuous and the formula

$$f(x)=F'(x)=\frac{1}{2\pi}\int_{-\infty}^{\infty}e^{-itx}\phi(t)\,dt$$

expresses its density $f(x)$ in terms of the characteristic function.

Proof: Lukacs (1970, p.31-33)

Theorem 6.4 Convolution theorem

A distribution function F is the convolution of two distributions F_1 and F_2, that is

$$\begin{aligned}F(z)&=\int_{-\infty}^{\infty}F_1(z-x)\,dF_2(x)\\&=\int_{-\infty}^{\infty}F_2(z-x)\,dF_1(x)\\&=F_1\otimes F_2\end{aligned}$$

if, and only if, the corresponding characteristic functions satisfy the relation

$$f(t)=\phi_1(t)\phi_2(t).$$

Proof: Lukacs (1970, section 3.3, p.36)

The following definition and three theorems are particularly useful for implementation of the algorithm used in the numerical determination of the loss distribution. These results are omitted from the presentation of DeBuysscher & Szegö (unpublished) and Laurant & Gregory (2003).

Definition 6.5

The support of the probability mass function of the discrete distribution of X is the subset, $x \in \mathbb{R}$ such that $p(x) > 0$. Let this subset be denoted by S. This implies that $p(x) = 0$ for any $x \notin S$.

(e.g. Paolella, 2006, p.116)

Definition 6.6

A discrete probability distribution is said to be of the lattice type if, and only if, its support is in an arithmetical progression. It is completely atomic (i.e. discrete) with all the atoms located at points of the form $\{a + jh\}$ where a is real, $h > 0$, and j ranges over a certain nonempty set of integers. The characteristic function of a distribution of lattice type is given by

$$f(t) = e^{ait} \sum_{j=-\infty}^{\infty} p_j e^{jait}$$

where $p_j \geq 0$ and

$$\sum_{j=-\infty}^{\infty} p_j = 1$$

(Chung, 2001).

Theorem 6.5

A characteristic function is that of a lattice distribution, if and only if, there exists a $t_0 \neq 0$ such that $|f(t_0)| = 1$.

Proof: (Chung, 2001, p. 183)

Theorem 6.6

There is an $h > 0$ such that $|f(h)| = 1$ and $|f(t)| < 1$ for $0 < t < h$. In this case X has a lattice distribution with span $2\pi/h$.

Proof: (Durret p.131)

Theorem 6.7

Let the random variable X have a lattice distribution with possible values of the form $a + kh \; (k = 0, \pm 1, \pm 2, ...)$. Then

$$P(X = a + kh) = \frac{h}{2\pi} \int_{-\pi}^{\pi} \exp\{-iu(a + kh)\} f(u)\, du \tag{6.1}$$

for every integer k, where $f(u)$ is the characteristic function of the random variable X.

(Proof: Petrov, 1975 p. 12 or Feller, 1971, p. 511)

There are two cases worthy of highlighting. If $a = 0$ and h is any real number then the distribution is termed 'arithmetric' (Feller, 1971, p. 138). A special case of the arithmetic distribution is the case $a = 0$ and $h = 1$. This is called the integer lattice distribution (e.g. Chung, 2001, p.184). The characteristic function of such a random variable X has period 2π and the following inversion formula holds for each integer k:

$$P(X = k) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \exp(-iuk) f(u)\, du .$$

Common discrete distributions such as the binomial and Poisson distributions are integer lattice distributions.

6.3.1.2 Characteristic functions and credit loss distributions

Assume a portfolio with the following features:

n	: number of assets in the portfolio;
N_k	: nominal value of the *k*th asset;
$X_k = 1_{\{\tau_k \le t\}}$	: default indicator for asset *k* at time *t*;

R_k : recovery rate for asset *k*;

$M_k = N_k(1-R_k)$: exposure at default on the *k*th name;

$L(t) = \sum_{k=1}^{n} M_k X_k$: accumulated loss at time *t*.

The characteristic function, conditional on a particular state V, is independent. For a given state of the economy $V = v$, the conditional Fourier transform of the portfolio loss distribution is:

$$\begin{aligned} f_{L(t)|V=v}(u) &= \mathbb{E}\left[e^{-iuL(t)}\right] \\ &= \mathbb{E}\left[e^{-iu(M_1X_1+M_2X_2+\ldots+M_NX_N)}\right]. \end{aligned}$$

At time t, $P\left(X_k = 1|V\right) = p_t^{k|V}$ or $P\left(X_k = 0|V\right) = 1 - p_t^{k|V}$. Therefore,

$$\begin{aligned} \mathbb{E}\left[e^{-iM_kX_ku}|V\right] &= 1 - p_t^{k|V} + p_t^{k|V} e^{iM_ku} \\ &= 1 + p_t^{k|V}\left(e^{iM_ku} - 1\right). \end{aligned}$$

Here we are assuming fixed recovery rates and hence fixed loss given defaulted amounts. Therefore, by independence of the state of the economy, the unconditional characteristic function of the accumulated loss at time t is:

$$\begin{aligned} f_{L(t)}(u) &= \mathbb{E}\left[e^{iL(t)u}\right] \\ &= \mathbb{E}\left[\prod_{k=1}^{n} 1 + p_t^{k|V}\left(e^{iM_ku} - 1\right)\right] \\ &= \int_{-\infty}^{\infty} \prod_{k=1}^{n} \left\{1 + p_t^{k|V}\left(e^{iM_ku} - 1\right)\right\} \varphi(V)\, dV, \end{aligned}$$

where $\varphi\left(V\right)$ is the standard normal density function.

The above integral is evaluated using the Gauss-Hermite quadrature technique. Thus, using K nodes,

$$f_{L(t)}(u) = \sum_{j=1}^{K} \left\{ 1 + p_t^{k|x_j} \left(e^{iM_k u} - 1 \right) \right\} w_j \,, \tag{6.2}$$

where w_j and x_j are the weights and abscissas respectively.[6]

By theorems 6.5 and 6.6, it may be confirmed that this characteristic function is of lattice type with period $2\pi/h$.

6.3.1.3 Numerical determination of the inverse characteristic function

The inverse characteristic function of a distribution of lattice type is the probability mass function (pmf). The pmf is obtained as the inverse Fourier transform of the characteristic function in equation (6.2). The pmf of the loss function is given by

$$P\left(L(t) = kh\right) = \frac{h}{2\pi} \int_{-\pi/h}^{\pi/h} \exp\left(-iukh\right) f_{L(t)}(u)\,du \tag{6.3}$$

The integral in equation (6.3) may be determined directly by numerical integration. This however is computationally inefficient. The FFT routine is a particularly efficient at the computation of Fourier transforms. Direct numerical integration requires $O\left(N^2\right)$ operations whilst the FFT requires $O\left(N \log_2 N\right)$ operations to compute N pmf values (Mittnik et al, 1999). In order to use the FFT the integral in equation (6.3) needs to be discretised.

For a given time t we seek to determine the pmf at points $x_k = \left(k-1\right)h,\ k = 1, \ldots, N.$ Let $u = 2\pi\omega$, then equation (6.3) becomes

$$P\left(\left(k-1\right)h\right) = h \int_{-\pi/h}^{\pi/h} e^{-i2\pi(k-1)h} f_{L(t)}(2\pi\omega)\,d\omega \tag{6.4}$$

The integral in equation (6.4) can be approximated by the left-hand Riemann sum for N points with spacing s as follows:

[6] Appendix A provides details on the Gauss-Hermite quadrature.

$$P\left(\left(k-1\right)h\right) \approx hs\sum_{n=1}^{N} f_{L(t)}\left(2\pi s\left(n-1\right)\right)e^{-i2\pi(n-1)(k-1)hs}.$$

By setting $hs = N^{-1}$,

$$P\left(\left(k-1\right)h\right) \approx \frac{1}{N}\sum_{n=1}^{N} f_{L(t)}\left(2\pi s\left(n-1\right)\right)e^{-i2\pi(n-1)(k-1)/N} \tag{6.5}$$

The form of equation (6.5) is identical to that of the inverse FFT function of Matlab. The summation may be efficiently computed by applying the inverse FFT to the sequence

$$f_{L(t)}\left(2\pi s\left(n-1\right)\right), \quad n = 1,...,N.$$

6.3.2 Recursion

This approach was first presented by Andersen et al. (2003). As was the case for the numerical inversion of the characteristic function, conditional on a particular scenario, the individual assets in the portfolio are independent. This allows for the use of the 'exact' recursive method for determining the convolution. The method is called exact because the overall potential losses are some multiple of a discretised loss unit. Let u denote the loss unit and let the loss weight be given by $w(k) = \langle M_k/u \rangle$, where $\langle x \rangle$ denotes the nearest integer to x. Where the distribution of losses in a portfolio is of a lattice type, the rounding to the nearest integer would not be needed. The maximum portfolio loss is $L_{\max} = \sum_{k=1}^{n} M_k = u \cdot l_{\max}$, where $l_{\max} = \sum_{k=1}^{n} w(k)$. Andersen et al. (2003) provide a method to choose a loss unit subject to a discretisation error, when the distribution of losses is not of a lattice type. Their approach is given in Appendix B.

This technique has been used by actuaries in the context of the individual risk model (Panjer & Willmot, 1992). As in the previous section, the losses in the portfolio need to be of a lattice type.

The probability density function of the aggregate loss conditional on the scenario V at time t is given by

$$f_S\left(x|V\right) = f_{X_1} \otimes f_{X_2} \otimes ... \otimes f_{X_n}\left(x|V\right),$$

where

$$f_{X_j}\left(x|V\right) = \begin{cases} p_j^{t|V} = 1 - q_j^{t|V}, & x = 0, \\ \quad q_j^{t|V}, & x = b_j. \end{cases}$$

The amount b_j is the amount lost in the event of the *j*th entity defaulting. The above density can be calculated recursively over the partial sums $S_j = S_{j-1} + X_j$, for $j = 2,3,...,n$ beginning with $S_1 = X_1$.

$$\begin{aligned} f_{S_j}\left(x|V\right) &= \begin{cases} f_{S_{j-1}}\left(x|V\right) f_{X_j}(0) & x < b_j, \\ f_{S_{j-1}}\left(x|V\right) f_{X_j}\left(0|V\right) + f_{S_{j-1}}\left(x - b_j|V\right) f_{X_j}\left(b_j|V\right) & x \geq b_j, \end{cases} \\ &= \begin{cases} p_j^{t|V} f_{S_{j-1}}\left(x|V\right) & x < b_j, \\ p_j^{t|V} f_{S_{j-1}}\left(x|V\right) + \ q_j^{t|V} f_{S_{j-1}}\left(x - b_j|V\right) & x \geq b_j. \end{cases} \end{aligned} \tag{6.6}$$

Once the loss unit, u, is determined, the algorithm to calculate the loss distribution as multiples of the loss unit at time *t*, for each scenario as given by the number of nodes in the Gaussian quadrature, K, follows below, after the definition of the notation to be used is given.[7] Define a matrix A with dimensions $\left(l_{\max} + 1\right) \times n,$ where n is the number of assets in the portfolio. The rows of A represent the range of discretised losses that range from 0 to $l_{\max}$. The algorithm is as follows:

1. determine the conditional default probability for each asset;
2. the value of $A_{1,1}$ is initialised with the probability of no loss, whilst the value of $A_{j,1} = P\left(X = b_j\right)$, which is the probability of default for the asset determined in step 1, is such that the rest of the values are set to zero;
3. for assets $j = 2,...n,$ the recursion of equation (6.6) is used to add assets to obtain the conditional probabilities, this will be the column *n* of the matrix A;
4. the conditional probabilities (column *n* of the matrix *A*) are stored in another matrix, say *B*, with dimensions $\left(l_{\max} + 1\right) \times K;$
5. steps 2 to 4 are repeated K times;

[7] For practical purposes, this is taken to be the coupon dates of the CDO contract.

6. the unconditional probability for the loss amount, $l = 0,\ldots,l_{\max}$ is given by $\sum_{j=1}^{K} B(l,j) \times w_j$, where the w_j are the weights of the Gaussian quadrature.

6.3.3 Collective-risk-model approximation to the individual risk model

In practice the computational burden of large portfolios becomes onerous especially using the recursion technique outlined above. It has become popular (Klugman et al. p. 203) to approximate the distribution of losses in the individual risk model by using the compound Poisson distribution (CPD). The CPD can be calculated by a recursive procedure or by the FFT. This section begins with a description of the mathematics of the Poisson and compound Poisson distributions. This is followed by a description of the theory and numerical implementation of the Panjer recursion and FFT methods when determining the compound Poisson distribution.

6.3.3.1 Properties of the Poisson distribution

The probability function $\tilde{p}_k$ denotes the probability that exactly k events occur. Let N be a random variable representing the number of such events. Then

$$\tilde{p}_k = \Pr(N = k), \;\; k = 0,1,2,\ldots$$

The probability generating function (pgf) of a discrete random variable N with probability function $\tilde{p}_k$ is

$$P(z) = P_N\left(z^N\right) = \sum_{k=0}^{\infty} \tilde{p}_k z^k.$$

Of particular interest is the pgf of the Poisson distribution that is

$$\begin{aligned} P_X(z) &= \sum_{x=0}^{\infty} z^x \frac{\lambda^x e^{-\lambda}}{x!} \\ &= e^{-\lambda} \sum_{x=0}^{\infty} \frac{(z\lambda)^x}{x!} \\ &= e^{-\lambda} e^{z\lambda} \\ &= e^{\lambda(z-1)}. \end{aligned}$$

The mgf is $M_X(t) = P_X(e^t) = e^{\lambda(e^t - 1)}$.

6.3.3.2 The link between generating and characteristic functions

The theorem below provides the relationship between the characteristic function φ_X and the pgf P_X. It is especially useful when computing the conditional aggregate loss distribution via the FFT.

Theorem 6.8

If the probability generating function exists for a random variable X, then $P_X(z) = \varphi(-i \ln z)$ and $\varphi_X(z) = P(e^{iz})$.

(Proof: Panjer, 2006, p.146).

If the random variable X has a discrete distribution with values $x_1, x_2, ...,$ with associated probabilities $p_1, p_2, ...,$ then

$$\begin{aligned} f(t) &= E\left[e^{itx_n}\right] \\ &= \sum_n p_n e^{itx_n}. \end{aligned}$$

6.3.3.3 Compound model for aggregate losses

Let S denote the aggregate losses associated with a set of N observed losses $X_1, X_2, ..., X_N$ satisfying the assumption that, given that there are n losses, the loss sizes are mutually independent random variables whose common distribution does not depend on n. The random sum

$$S = X_1 + X_2 + ... + X_N$$

has distribution function

$$\begin{aligned} F_S(x) &= P(S \le x) \\ &= \sum_{n=0}^{\infty} p_n P(S \le x | N = n) \\ &= \sum_{n=0}^{\infty} p_n F_X^{*n}(x) \end{aligned} \tag{6.7}$$

where $F_S(x) = P(X \le x)$ is the common distribution of $X_j, j = 1,2,...,N$ and $p_n = P(N = n)$. F_X^{*n} is the 'n-fold convolution' of the cdf of X. It can be obtained as

$$F_X^{*0} = \begin{cases} 0 & x < 0, \\ 1 & 1 \ge 0, \end{cases}$$

and

$$F_X^{*k}(x) = \int_{-\infty}^{\infty} F_X^{*(k-1)}(x-y)\, dF_X(y) \text{ for } k = 1,2,... \tag{6.8}$$

In the case of discrete random variables with positive probabilities at 0,1,2,…, equation (6.8) reduces to

$$F_X^{*k}(x) = \sum_{y=0}^{x} F_X^{*(k-1)}(x-y) f_X(y) \text{ for } x = 0,1,..., \ k = 2,3,...$$

The corresponding probability function is

$$f_X^{*k}(x) = \sum_{y=0}^{x} f_X^{*(k-1)}(x-y) f_X(y) \text{ for } x = 0,1,..., \ k = 2,3,...$$

The distribution equation (6.7) is called a compound distribution, and the probability function for the distribution of aggregate losses is

$$f_S(x) = \sum_{n=0}^{\infty} p_n f_X^{*n}(x).$$

The pgf of S is given by

$$\begin{aligned} P_S(z) &= E\left[z^S\right] \\ &= \sum_{n=0}^{\infty} E\left[z^{X_1+X_2+\ldots+X_n} \,|N=n\right] P(N=n) \\ &= \sum_{n=0}^{\infty} E\left[\prod_{j=1}^{n} z^{X_j}\right] P(N=n) \\ &= \sum_{n=0}^{\infty} P(N=n)\left[P_X(z)\right]^n \\ &= E\left[P_x(z)^N\right] = P_N\left[P_X(z)\right]. \end{aligned} \tag{6.9}$$

$P_N(z)$ and $P_X(z)$ are referred to as the primary and secondary distributions, respectively.

The characteristic function is used in the next section. It is given by

$$\begin{aligned} \varphi_S(z) &= E\left(e^{izS}\right) \\ &= P_N\left[\varphi(z)\right]. \end{aligned}$$

6.3.3.4 Properties of the CPD

A recursive formula attributed to Panjer (1981) is the most widely used method of determining values for the CPD. Another method that is frequently used as the processing power of computers increases and its costs decrease, is numerical inversion by the FFT of the CPD characteristic function. In this section the above two methods will be described.

Panjer recursion

The probability of exactly k losses can be written as

$$\begin{aligned} P\left(S=k\right) &= \sum_{n=0}^{\infty} P\left(S=k \middle| N=n\right) \\ &= \sum_{n=0}^{\infty} P\left(M_1+M_2+\ldots+M_N=k \middle| N=n\right) P\left(N=n\right). \end{aligned}$$

Let $g_n = P\left(S=n\right), p_n = P\left(N=n\right)$ and $f_n = P\left(M=n\right)$. Then the above formula may be written as

$$g_k = \sum_{n=0}^{\infty} p_n f_k^{*n} ,$$

where $f_k^{*n}, k = 0,1,2...$ is the *n*-fold convolution of the function $f_k, k = 0,1,...$ This means that the probability that the sum of n independent and identically distributed random variables each with probability function f_k will take on the value k.

Definition 6.7 $(a,b,0)$ class of distributions

Let p_k be the probability mass function of a discrete random variable. It is a member of the $(a,b,0)$ class of distributions if constants a and b exist such that

$$\frac{p_k}{p_{k-1}} = a + \frac{b}{k}, \quad k = 1,2,3,... \tag{6.10}$$

(Panjer, 2006, p.115)

When $P_N(z)$ is a member of the $(a,b,0)$ class then the above relationship holds for its pmf.

Theorem 6.9 Panjer recursion

If the primary distribution is a member of the $(a,b,0)$ class then the recursive formula to calculate the compound distribution probabilities is given by

$$g_k = \frac{1}{1 - af_0} \sum_{j=1}^{k} \left(a + \frac{bj}{k} \right) f_j g_{k-j}, \quad k = 1,2,3,... \tag{6.11}$$

Proof: (Panjer, 2006, p.127)

If the primary distribution is Poisson, we have $a = 0,$ and $b = \lambda$, then the formula in equation (6.11) becomes

$$g_k = \frac{\lambda}{k} \sum_{j=1}^{k} j f_j g_{k-j},$$

with starting values

$$\begin{aligned} g_0 &= P(S=0) \\ &= P(0) \\ &= P_N\left[P_M(0)\right] \\ &= P_N(f_0) \\ &= e^{-\lambda(1-f_0)}. \end{aligned}$$

Distributions of this type are called compound Poisson distributions.

6.3.3.4.1 Numerical inversion of the characteristic function

Given a random sum $X_1 + X_2 + \ldots + X_N$ that has the distribution $U = \sum p_k F^{*k}$ with common characteristic function $\varphi(t)$, then the corresponding characteristic function for U is

$$\varphi_S(z) = P_N(\varphi_X(z)).$$

For the case in which the primary distribution is Poisson, we have

$$\varphi_S(z) = e^{\lambda(\varphi_X(z)-1)}.$$

(Feller, 1971, p. 504)

6.3.3.5 Literature on the numerical computation of the CPD

Prior to the advent of the FFT and the Panjer recursion technique, the determination of the CPD had traditionally been determined directly, an example of which is given in Klugman et al. (2004, p.145).

> The usual method of evaluating the distribution function requires the computation of many convolutions of the conditional distribution of the amount of a claim given that a claim has occurred.

(Panjer, 1981)

The first published article identified, during the course of this research, with respect to the computation of the CPD was that of Panjer (1981). The recursive technique was a major advance. Prior references to the inversion of the characteristic function by the FFT did not address the CPD directly. Bühlmann (1984) provided an analysis of the comparison of the FFT and Panjer recursion with respect to computational efficiency. Embrechts & Frei (2007) updates and extends the comparison to include aliasing error and exponential tilting.[8] Belov & Galkin (1991) reviews all the available algorithms and comments that the Panjer recursion is the simplest to implement.

6.3.3.6 Obtaining the pdf of a compound Poisson distribution

The probability distribution is obtained by inverting the characteristic function of the compound Poisson distribution. To test the code the example from Klugman et al. (2004, p. 187, Example 6.28) was implemented in Matlab.[9]

Example: Suppose the random variable X takes on values 1, 2 and 3 with probabilities 0.5, 0.4 and 0.1 respectively. Further suppose the number of claims has the Poisson distribution with parameter $\lambda = 3$. Use the FFT to obtain the distribution of S using n=8 and n=4096.

[8] In addition the authors provide examples of the CPD implemented in the R programming language using the *actuar* package.

[9] The program implemented in Matlab is as follows:

```
M=4096; f1=[0,0.5,0.4,0.1,zeros(1,M-4)];
fhat1=fft(f1);
P1=exp(3*(fhat1-1));
g1=ifft(P1);
```

Table 6.1 Published and calculated values of the CPD using the FFT

	f_S	Matlab implementation
	n = 4096	n = 4096
0	0.04979	0.04979
1	0.07468	0.07468
2	0.11575	0.11575
3	0.13256	0.13256
4	0.13597	0.13597
5	0.12525	0.12525
6	0.10558	0.10558
7	0.08305	0.08305

6.3.3.7 Valuation of a synthetic CDO portfolio using the CPD

De Prisco et al. (2005) first introduced the CDP to the pricing of CDO tranches. In their implementation the entire loss distribution is not calculated. The implementation in this research differs from this approach. The other methods considered all determine the loss distribution so an equivalent comparison can be made.

6.3.4 Summary

The numerical inversion of the characteristic function and the recursion technique yielded identical results in Table 6.2 below, based on the example given in chapter 5. This was also the case for the CPD computed via recursion and using the Panjer recursion.

Table 6.2 The tranche spreads in basis points.

Tranche attachement points	Default-time	Numerical inversion	Exact recursion	CPD	
	MC	FFT	Exact	Panjer	FFT
0–25	12.90095	12.67298	12.67298	12.55555	12.55555
25–75	3.60897	3.59998	3.599979	3.57876	3.57876
75–150	0.91889	0.91665	0.916652	0.91621	0.91621
150–400	0.04970	0.04992	0.049917	0.05208	0.05208

The results presented in Table 6.3 indicate that the Panjer recursion is eclipsed in terms of the computational efficiency by the FFT method. The recursive method is competitive purely because there are fewer 'for loops'[10] in its implementation. Ideally the computer code should therefore be written in a lower-level language such as C. Another issue with the Panjer recursion is that of underflow or overflow which occurs when the initial probability is too small to be represented in a computer. Solutions to this problem have been addressed in Panjer & Willmot (1986).

Table 6.3 The time in seconds for each tranche spread calculation.

Default-time	Numerical inversion	Recursion	CPD	
Monte Carlo	FFT	Exact	Panjer	FFT
15.98	1.36	0.73	3.13	0.40

[10] Matlab is particularly slow at executing 'for loops'. A possible solution is to remove 'for loops' via a process called 'vectorization'.

Table 6.4 Relative errors with respect to Monte Carlo simulation estimates

Tranche attachment points		Recursion	CPD
Lower	Upper	Exact	FFT
0	25	1.767%	2.677%
25	75	0.249%	0.837%
75	150	0.244%	0.292%
150	400	0.439%	4.788%

Table 6.4 indicates that the CPD does not perform well when pricing the junior and senior tranches.

6.4 Numerical experiments

The portfolio used thus far is of a lattice type as per Definition 6.6. This is not the case for most portfolios. The semi-analytic methods presented in this chapter is applicable when the distribution of losses is of a lattice type. Andersen et al. (2003) provided an algorithm to transform the distribution of losses to a lattice type subject to a chosen error. The algorithm is reproduced in Appendix B.

In this section, contrived portfolios are used to evaluate the accuracy and computational efficiency of the respective algorithms. Two portfolios are used. The numbers of names in these two portfolios are 125 and 250 respectively. The recovery rate increases linearly, for both portfolios, from 10–30%. All the names in the portfolio have a common notional amount of 100 and a constant default probability of 220 basis points over the five-year term. Given that the recovery rate is different for each entity in the portfolio the hazard rate will also be different. This is determined using the method described in chapter 2. The loss unit used was five.

For both portfolios, a Monte Carlo simulation was run 100 times where each simulation consisted of 100 000 iterations. The mid point of the 95th percentile was chosen as the value for the tranches. The following tranches were evaluated: 0–500, 500–2500, 2500–5000 and 5000–10000.

Table 6.5 The fair tranche spread and the computational times for the portfolio consisting of 125 names

Tranche attachment points	Monte Carlo	Recursion	Numerical inversion	CPD
0–500	21.86507	21.17588	21.17579	21.12751
500–2500	6.55731	6.62159	6.62159	6.56823
2500–5000	1.89889	1.83851	1.83851	1.82446
5000–10000	0.27605	0.26534	0.26534	0.271015
Time (seconds)	21.11220	6.51059	34.15806	0.53831

Table 6.6 The fair tranche spread and the computational times for the portfolio consisting of 250 names

Tranche attachment points	Monte Carlo	Recursion	Numerical inversion	CPD
0–500	30.24512	29.10335	29.10335	29.06861
500–2500	11.13897	10.90346	10.90346	10.88929
2500–5000	4.879363	5.120008	5.120008	5.07228
5000–10000	1.881993	1.80306	1.80306	1.79583
Time (seconds)	36.32774	25.00510	60.46545	0.80254

Table 6.7 The relative errors of the recursion and CPD methods with respect to Monte Carlo simulation estimates for the 125 and 250 name portfolios

125		250	
Recursion	CPD	Recursion	CPD
3.152%	3.373%	3.775%	3.890%
0.980%	0.167%	2.114%	2.241%
3.180%	3.920%	4.932%	3.954%
3.882%	1.825%	4.194%	4.578%

The method using the inversion of the characteristic function is omitted given that it was slower than the recursive method, and it produces values that are identical to the recursive method. Tables 6.5–6.7 present the results of using the numerical inversion of the characteristic function, the recursive and compound Poisson approximation.

6.5 Conclusion

The CPD evaluated using the FFT proved to be the most efficient (as per the bottom row of Tables 6.5 and 6.6) with a loss of accuracy (as per Table 6.7) that is modestly higher than that produced by the next most competitive method; that of Andersen et al. (2003). The Panjer evaluation of the CPD proved computationally too demanding. This was also the case for the numerical inversion of the characteristic function via the FFT. Increasing the number of names in the portfolio diminishes the performance of the recursive method.

The CPD evaluated using the FFT proved to be the most efficient (as per the bottom row of Tables 6.5 and 6.6) with a loss of accuracy (as per Table 6.7) that is modestly higher than that produced by the next most competitive method; that of Andersen et al. The Panjer evaluation of the CPD proved computationally too demanding. This was also the

case for the numerical inversion of the characteristic function via the FFT. Increasing the number of names in the portfolio diminishes the performance of the recursive.

Chapter 7

Conclusion

This research has considered the most cited models in the literature that are being used to determine the credit loss distribution and value of synthetic CDO tranches in the financial industry. Two major omissions in the literature have been identified. These are an overall comparison of the models with respect to computational efficiency and accuracy and the elucidation of their algorithms. This was the primary research objective of this research. A secondary objective was to highlight the contribution of actuarial science to the credit risk valuation field.

This chapter is organised as follows. The first four sections review and summarise the results of this work. The extensions to the models are also elaborated on. In the fifth section, topics for future of research are suggested. The events in the financial markets since August 2007, which have caused major losses for the financial institutions, revealed that the models used have been inadequate. The chapter concludes with a discussion on the financial crisis and the shortcomings of the models discussed in this dissertation.

7.1 Obtaining default probabilities

A method for obtaining default probabilities from market quotes of CDSs was illustrated in chapter 3. This method utilises the survival-function approach which is familiar to actuaries to obtain the hazard rate. The hazard rate is then converted to a default probability. A common approximation method was tested against the accurate method and it was established that this satisfactorily estimated the default probability. The approximation described in this chapter was used throughout this research.

7.2 Multi-step approach

The multi-step approach could also be described as the first-passage-time model applied to successive intervals. The use of this approach for the valuation of CDOs has two steps: the determination of the default thresholds and the loss distribution.

This research tested all the published methods of determining the default threshold with respect to accuracy and computational efficiency using a contrived example. The methods tested are Hull & White (2001) (an analytic approximation), Iscoe & Krenin (unpublished a) (a simulation technique) and Morokoff (2003) (a convolution method that is approximated by the FFT). For each of these methods, the key parameters are the number of grid points, the number of simulations and the number of nodes. In addition, the Monte Carlo method was enhanced using antithetic variates. For all the methods tested, the parameters were chosen so that no additional accuracy was observed by increasing the parameter values (which would increase the computational time). The methods were discussed and their algorithms exposed. Given that Morokoff (2003) omitted implementation details, the exposition given may increase the usage of the convolution technique.

The method of Iscoe & Krenin (unpublished a) was found to be slower but more accurate than that of Hull & White (2001). The method that used the FFT was found to be computationally efficient and accurate. A further observation made was that different methods used to determine the default thresholds yielded different results with respect to accuracy. Morokoff (2003) proved to be the most accurate.

The method of Morokoff (2003) was then used as the benchmark to price synthetic CDO tranches. For the example used, it was found that CDO tranche prices using Hull & White (2001) produced errors that increased from approximately 1% for the equity tranche to 3% for the senior tranche. The range for the method of Iscoe & Kreinin (unpublished a) was approximately 0.5 to 1.4%.

7.3 Default time

It was shown that the default-time approach improved the computational efficiency of determining the prices of the tranches of CDOs when compared to that of the multi-step approach. However, there were significant differences in their respective credit loss distributions in the tail regions.

7.4 Semi-analytic methods

The efficiency of the default-time approach and its wide spread use made it the focus of many researchers who sought to develop semi-analytic approximations. This was done primarily for computational efficiency. Different techniques were implemented to assess the accuracy and computational efficiency of the semi-analytic methods with respect to the standard Monte Carlo implementation of the default-time model.

The methods that were implemented are numerical inversion of the characteristic function, exact recursion and the compound Poisson approximation. The compound Poisson approximation was implemented using the Panjer recursion and the FFT method.

In addition, the algorithms for these methods were described. In all the published literature, none of the algorithms for the methods considered are published. The implementation of the numerical inversion of the characteristic function using the FFT proved particularly difficult. The standard reference used for the numerical inversion of characteristic function in the credit-risk context, DeBuysscher & Szegö (unpublished), omits theorems that are essential to its understanding and successful implementation. This is ameliorated by the presentation given in chapter 6. The other semi-analytic methods analysed proved easier to implement.

The compound Poisson approximation, implemented using the FFT, was seen to be the computationally most efficient though it was the least accurate. The Panjer recursion implementation of the compound Poisson approximation proved to be computationally inefficient. This was the case for a 50-entity portfolio, and hence was not considered for larger portfolios that were subsequently examined. For the heterogeneous portfolio that was not of a lattice type, the numerical inversion of the characteristic function and the exact recursion method produced identical results when a common loss unit was used.

All the methods were implemented in Matlab. This was the main reason why the numerical inversion of the characteristic function was slower than the exact recursion method. This is because Matlab is slower at executing ‘for loops’.

The findings confirm that of Glasserman & Ruiz-Mata (2006) that the accuracy and computational efficiency of the Monte Carlo method is still preferred when computational time is fixed.

7.5 Future research

This research investigated the computational efficiency and accuracy of the most cited models pertaining to the valuation of the credit loss distribution and the pricing of synthetic CDO tranches. As stated above Matlab's major deficiency is its implementation of for loops. An invaluable exercise would be to implement these models in a lower-level language such as C or C++. This might improve the performance of the FFT method.

The models discussed for this research in the context of credit risk may be considered 'static'. The default correlations and probabilities are constant. In the literature 'dynamic' models have appeared. These models extend the Heath-Jarrow-Morton model (1992) used to value interest-rate financial instruments to the valuation of credit risk. These models are not as mathematically tractable as the ones considered for this research. An example of this type of model is presented in Bennani (2006).[11] A key feature of models actually used is that they are easy to implement. These models do not fall into this category.

A topic for further investigation is to assess whether other variance-reduction techniques could be applied to the inverse of the default boundary problem as presented by Iscoe & Kreinin (unpublished a). Arvanitis et al. (unpublished) discuss the use of importance sampling to improve the computational efficiency in determining the credit loss distribution.

Further research into the use of the CPD used for credit risk may entail improving the approximation by reducing the errors. Schaller & Temnov (2008) compares techniques that reduce errors when determining convolutions particularly of compound distributions.

This research also highlighted the significant contribution of actuarial science to the credit-risk valuation field. The use of the conditional independence framework introduced in chapter 6 may have use in actuarial science, for example, in the valuation of insurance contracts that depend on a systemic event.

[11] Bennani, N. "The Forward Loss Model: A Dynamic Term Structure Approach for the Pricing of Portfolio Credit Risk," Working Paper, The Royal Bank of Scotland (2006)

7.6 Models and the financial crisis of 2007 – 2008

This research introduced the major models that were used to evaluate the credit risk of entities in a portfolio. In the light of the financial crisis and the complicity of these types of models used in the valuation of portfolio credit risk, the models discussed in this research have been shown to be inadequate. The shortcomings of the models are:

- the use of constant correlation;
- the use of continuous processes;
- assuming that asset returns are Gaussian;
- the absence in the models of any input from behavioural finance.

The financial crisis had many other causes, but the following appear most frequently in the media:

- the asymmetry of information between buyer and seller;
- the effects of contagion;
- the complicity of regulators[12] (The Economist, October 2008).

The asymmetries of information led to a widespread breakdown of trust in the money market. Trust is the—if not the only—key ingredient of the banking industry. Seabright (2004)[13] provided a plausible reasoning as to the origins of banking and the importance of trust in banking:

> Borrowing and lending are as old as communal living, as deeply embedded in social life as the sharing of meat from a kill. But banks are a remarkable and much more recent innovation; no records can tell us when they first began, but a plausible guess is that it was after the invention of agriculture and before, not after, the invention of money. They may have begun simply as storehouses (certainly that was true of the first recorded banks). Their subsequent transformation into proper banks may even have been the fruit of deceit, the ingenuity of a warehouse owner who realized that he could lend out some of the grain he was

[12] Beddoes, Z.A., (2008), The World Economy Special Report, *The Economist*, 11 October

[13] Seabright, P. (2004) *The Company of Strangers*, Princeton University Press, New Jersey

> storing on others' behalf without the knowledge of its owners.

However as Frydman, Goldberg & Phelps (2008)[14] state:

> In searching for clues to the financial disaster, attention has focused on predatory practices in originating mortgages and on investment banks' 'cosy' relationship with the agencies entrusted to rate their structured assets. But there is another cause: failure to acknowledge that market participants – including financial institutions – and regulators alike have only imperfect knowledge about the forces and mechanisms driving asset values and the broader economy.

Greenspan (2008)[15] remarks that the task to quantify financial risk is unattainable:

> We will never be able to anticipate all discontinuities in financial markets. Discontinuities are, of necessity, a surprise. Anticipated events are arbitraged away. But if, as I strongly suspect, periods of euphoria are very difficult to suppress as they build, they will not collapse until the speculative fever breaks on its own. Paradoxically, to the extent risk management succeeds in identifying such episodes, it can prolong and enlarge the period of euphoria. But risk management can never reach perfection. It will eventually fail and a disturbing reality will be laid bare, prompting an unexpected and sharp discontinuous response.

The complicity of the regulators occurred in a general environment of deregulation where it was adjudged that the market would be able to regulate itself with minimum intervention by governments. Regulators believed that credit risk would best be managed if it were allowed to be dispersed throughout the financial system, the so-called 'originate and distribute' model. Also, the ratings of the credit rating agencies were codified into the Basel capital regulations. The well known conflict of interest between the rating agency and their paymasters (the originators) was left unresolved. The highest rated assets, those rated AAA say, were given a low capital weighting. They were believed to be almost risk-free. The opposite was true of lowly rated assets. The industry sought to repackage

[14] Frydman, R., Goldberg M. & Phelps, E, (2008) We must not rely on the rosiest ratings, *Financial Times*, 19 October

[15] Greenspan, A (2008) We will never have a perfect model of risk, Financial Times, 16 March

their balance sheets into CDOs and sell the lowest rated tranche. The most highly rated tranches were kept on the financial institutions' balance sheets as they attracted the least capital charge. This worked for a brief period until the repackaged assets such as those CDOs containing United States subprime mortgages began to default. As the number of defaults increased, the senior tranches of CDO structures began to suffer losses as well. Those institutions that invested or insured certain tranches of CDOs also suffered losses, such as American Insurance Group. As the situation became apparent to the rating agencies, they started to downgrade CDO tranches and institutions that sold credit protection on CDO tranches. This resulted in an increase in the capital requirement of financial institutions, which in turn left less capital available to fund their business. When most financial institutions reduced or stopped lending to each other, the interbank funding market seized. This was termed the 'credit-crunch'.[16] The crisis began to be first noticed in the money market. Other parts of the financial markets were soon also affected due to the contagion.

Derman (2005)[17] warned about the use of mathematical models in finance and accused the economics profession of 'physics envy': an overzealous desire for mathematical rigor:

> But economists seem to have embraced formality and physics envy without the corresponding benefits of accuracy or predictability. In physics, Maxwell's theory and quantum mechanics allow you to predict the way an electron spins about its own axis inside a hydrogen atom to an accuracy of 12 decimal places. Something that accurate isn't just a model—it's a law. In economics, by contrast, there are no laws at all, only models, and you're immensely lucky if you can predict up from down.

Seabright (2004) provided a prescient warning of the fragility and faith that is placed with financial institutions:

> But lending money is risky in its own way. Its anonymity and the ease with which it can be concealed make fraud all the more tempting. So banks have not completely displaced

[16] e.g., Tett, G (2008) Big Freeze part 1: How it began, *Financial Times*, 3 August

[17] Derman, E (2005) Beware of Economists Bearing Greek Symbols, *Harvard Business Review*, October

> that old standby of peasant societies, the storeroom in the comer of your house. Over a billion peasants around the world will store food at the end of the harvest for as long as the supply holds out. Even those who sell produce often feel the money is less safe in a bank than under a mattress. In rich industrial countries it *is* safer in the bank—most of the time. For banks have flourished because of their ingenious capacities for helping people to live with risk. Like many of the human institutions that do so, they have reduced everyday risks to levels low enough to make us forget that risk exists at all, sometimes leaving us startlingly unprepared for the more unusual hazards, whose impact can be very large.

Thomson (2004) distinguishes between two types of science, the traditional and the modelling approach. In this framework, the physics that Derman (2005) refers to would fall into the traditional approach. Thomson (2004) defines actuarial science as a science which adopts a modelling approach to decision making with the use of professional judgement.

This offers a way forward for financial economics, which has been rooted in the frequentist approach to probability theory. The financial crisis has forced a reassessment of the mathematics and the assumptions used, e.g. that asset returns are normally distributed (a criticism of this assumption and other assumptions used in finance is given in Taleb (2007)[18]). The key elements that are at the heart of the financial crisis which were not captured by the existing models used in finance—such as imperfect knowledge and euphoria—may be accommodated by a Bayesian approach to modelling and the use of behavioural finance. In the Bayesian approach, the prior distribution would need to incorporate the contingencies that were not previously considered. The use of the actuarial control cycle, being an application of the Bayesian approach, would seem the most appropriate path forward.

[18] Taleb, N (2007) *The Black Swan*, Random House, New York

Glossary

CDO	Collateralised debt obligation, p.3.
CDS	Credit default swap, p.2.
CPD	Compound Poisson distribution, p. 83.
DFT	Discrete Fourier transform, p. 29.
FFT	Fast Fourier transform, p. 30.
SPV	Special purpose vehicle, p. 3.

Appendix A
An introduction to Gauss-Hermite quadrature

The portfolio Fourier transform (or characteristic function) requires the computation of improper integrals such as:

$$\int_{-\infty}^{\infty} g(x)\phi(x)dx$$

where $\phi(x) = \frac{1}{\sqrt{2\pi}} \exp\left(\frac{-x^2}{2}\right)$.

This integral does not have an analytic expression and must be computed numerically. A typical approach would be to use formula like:

$$\int_{-\infty}^{\infty} h(x)dx \approx \Delta \sum_{j=0}^{N} h(x_j)$$

where $x_j = x_0 + j\Delta,\ j = 0,1,...,N$ for $h(x) = g(x)\phi(x)$. This approach however is computationally inefficient when compared with the Gaussian quadrature approach (DeBuysscher, et al. 2003).

The Gaussian quadrature consists in approximating the integral:

$$\int_{-\infty}^{\infty} g(x)W(x)dx$$

where $W(x)$ is the weight function and is known, by the formula:

$$\int_{-\infty}^{\infty} g(x)W(x)dx \approx \sum_{j=0}^{N} g(x_j)w_j,$$

where the $x_j,\ j = 0,1,2,...,N$ are appropriately chosen abscissas and the $w_j,\ j = 0,1,...,N$ are the corresponding appropriate weights. The abscissas and weights have been determined for different functions $W(x)$. Of particular interest is the Hermite function $W(x) = e^{-x^2}$. For this function Abramowitz & Stegun (1972, Table 25.10) has published values for up to the first twenty polynomials. Chapter 6 of Fusai & Roncoroni (2007) provides an accessible introduction to quadrature methods.

Appendix B
Choosing the loss unit

If all losses in default for individual names are some multiples of the loss unit, there will be no approximation error arising from the discretisation of the loss distribution. In practice, however, this will not be the case.

Andersen et al. (2003) provide the following algorithm to determine a suitable loss unit subject to an error ε. This is to be chosen. When there is no approximation error the algorithm is identical to that of the greatest common divisor.

For a given loss unit u the error for a loss W_k is given by

$$\left|W_k - w_k u\right|$$

where w_k is the ratio W_k/u rounded to the nearest integer. If W is the smallest loss value $(>\varepsilon)$, define $u_n = W/n, n = 1,2,3,...$ and $w_k^{(n)}$ as the integer rounded ratio W_k/u_n. Let $\tilde{n}$ be the smallest integer such that $\left|W_k - w_k^{(n)} u_n\right| < \varepsilon$ for all $k.$ Then the loss unit is

$$u = \frac{\sum_{k=1}^{N} W_k}{\sum_{k=1}^{N} w_k^{(n)}}.$$

Appendix C
Computer Code

This section contains the listings of the program code used in this dissertation. The following two programs were not written by the author: consolidator.m and qrule.m. They are listed in the appropriate sections. The rest of the programs were written by the author.

C.1 Extraction of default probabilities from market CDS quotes

```
function y= NonLinearEqnsCDS(x)

% The inputs are:
%  delta:         The daycount convention. It is taken to be constant
%                 between coupon payment dates.
% Spreads:        A vector of the spreads.
% n:              The number of coupon payments (assuming no default).
% m:              The term of the contract in a smaller time-step than
%                  delta.
% x:               The initial values (guess) for each of the piecewise constant
%                  hazard rates to be used in the Newton solver.
%
% Output:      The piecewise constant hazard rates.
% Program Dependencies:
%  piecewise.m
%  newtons.m
%The newtons.m code is as per
% Yang, W.Y, Cao, W, Chung, TS & Morris, J. (2005) "Applied Numerical
% Methods using Matlab", John Wiley & Sons, Inc , New Jersey

delta =0.5;
Rec=0.3;
r=0.05;
spreads=[0.005,0.006,0.008,0.0105,0.0120,0.0140];

y(1)= npvCDS(delta,spreads(1),1,182,x,r,Rec);
y(2)= npvCDS(delta,spreads(2),2,365,x,r,Rec);
y(3)= npvCDS(delta,spreads(3),6,1095,x,r,Rec);
y(4)= npvCDS(delta,spreads(4),10,1825,x,r,Rec);
y(5)= npvCDS(delta,spreads(5),14,2555,x,r,Rec);
y(6)= npvCDS(delta,spreads(6),20,3650,x,r,Rec);


function y= npvCDS(delta,spread,n,m,x,r,Rec)

% This program calculates the Net Present Value of a Credit Default Swap
% Inputs: delta - The day count convention-assumed constant for the
%                   contract.
%            spread- scalar, specific to the maturity of the term of the
%                    market premium of the CDS
%
%            n- The number of coupon payments under the CDS assuming
%                 no-default.
%            m- The protection leg discretization frequency. Assumed
%                  daily.
%             r- risk free interest rate, assumed constant
%             Rec- The recovery rate.
%-----------------------------------------------------------------------------------------

y1=0;

for i = 1:n

    y1 = delta*spread*exp(-i*delta*r-piecewise(i*delta,x))+y1;

end
```

```
y2=0;
for j=1:m

   y2=  -delta*spread*(exp(-piecewise((j-1)/365,x))-exp(-piecewise(j/365,x)))*exp(-
r*j/365)*((j/365)-ceil((j/365)/delta)*delta)/delta...
      +(1-Rec)*(exp(-piecewise((j-1)/365,x)) - exp(-piecewise(j/365,x)))*exp(-r*j/365)+y2;
end

y=y2-y1;



function hh=piecewise(t,x)
%-------------------------------------------------------------------
% This is the implementation of the piecewise hazard rate function.
% The inputs are:
%      t     Time (scalar)
%      x     The hazard  rate (vector). Input due to initial starting
%            values.
% The method implemented does away with if statements.
%--------------------------------------------------------------------

hh = (t>=0 && t<=0.5).*x(1) *t + (t>0.5 && t<=1).*(0.5*x(1)+ x(2)*(t-0.5))...
      +(t>1 && t<=3).*(0.5*(x(1)+x(2))+x(3)*(t-1))...
      + (t>3 && t<=5).*(0.5*(x(1)+x(2))+2*x(3)+x(4)*(t-3))...
      +(t>5 && t<=7).*(0.5*(x(1)+x(2))+2*x(3)+2*x(4)+x(5)*(t-5))...
      + (t>7 && t<=10).*(0.5*(x(1)+x(2))+2*x(3)+2*x(4)+2*x(5)+x(6)*(t-7));


function [x,fx,xx] = newtons(f,x0,TolX,MaxIter,varargin)
%newtons.m to solve a set of nonlinear eqs f1(x)=0, f2(x)=0,..
%input: f = 1^st-order vector ftn equivalent to a set of equations
% x0 = the initial guess of the solution
% TolX = the upper limit of |x(k) - x(k - 1)| % MaxIter = the maximum # of iteration
%output: x = the point which the algorithm has reached
% fx = f(x(last))
% xx = the history of x
h = 1e-4; TolFun = eps; EPS = 1e-6;
fx = feval(f,x0,varargin{:});
Nf = length(fx); Nx = length(x0);
if Nf ~= Nx, error('Incompatible dimensions of f and x0!'); end
if nargin < 4, MaxIter = 100; end
if nargin < 3, TolX = EPS; end
xx(1,:) = x0(:).'; %Initialize the solution as the initial row vector
fx0 = norm(fx); %(1)
for k = 1: MaxIter
dx = -jacob(f,xx(k,:),h,varargin{:})\fx(:);%/;%-[dfdx]ˆ-1*fx
for l = 1: 3 %damping to avoid divergence %(2)
dx = dx/2; %(3)
xx(k + 1,:) = xx(k,:) + dx.';
fx = feval(f,xx(k + 1,:),varargin{:}); fxn = norm(fx);
if fxn < fx0, break; end %(4)
end %(5)
if fxn < TolFun | norm(dx) < TolX, break; end
fx0 = fxn; %(6)
end
x = xx(k + 1,:);
if k == MaxIter
   fprintf('The best in %diterations\n',MaxIter)
end
```

```
function g = jacob(f,x,h,varargin) %Jacobian of f(x)
if nargin < 3, h = 1e-4; end
h2 = 2*h; N = length(x); x = x(:).'; I = eye(N);
for n = 1:N
g(:,n) = (feval(f,x + I(n,:)*h,varargin{:}) ...
-feval(f,x - I(n,:)*h,varargin{:}))'/h2;
end
```

C.2 Determining the default threshold by direct integration

```
function DirectInteg1

a=norminv(0.05)
options = optimset('Display','iter','TolX',1e-8);
b=fzero(@(t)h1(t,a),a,options);
c=fzero(@(t)h2(t,b,a),b,options)
vpa(c)
end

function y=f1(x)
   y = normpdf(x);
end

function y= h1(t,a)
 y = quadgk(@(x)f1(x).*normcdf(t-x),a,Inf)-0.05;
end

function z=f2(x,a)
    n=length(x);  z=zeros(size(x));
    for j=1:n
        h=@(u)(f1(u).*normpdf(x(j)-u));
        z(j)=quadgk(h,a,Inf,'RelTol',1e-12,'AbsTol',1e-12);
    end

end

function y =h2(t,b,a)
 integrand = @(x,y) f1(x).*normpdf(t-x)*normcdf(t-y);
  y = dblquad(integrand,a,Inf,b,Inf,'quadgk')-0.05;
 end

 function z=f3(x,b,a)
    n=length(x);  z=zeros(size(x));
    for j=1:n
        h=@(u)(f2(u,a).*normpdf(x(j)-u));
        z(j)=quadgk(h,b,Inf,'RelTol',1e-12,'AbsTol',1e-12);
    end
 end

function y =h3(t,c,b,a)
 y= quadgk(@(u)(f3(u,b,a).*normcdf(t-u)),c,Inf)-0.05;
end

 function z=f4(x,c,b,a)
    n=length(x);  z=zeros(size(x));
    for j=1:n
        h=@(u)(f3(u,b,a).*normpdf(x(j)-u));
        z(j)=quadgk(h,c,Inf,'RelTol',1e-12,'AbsTol',1e-12);
    end
end

function y =h4(t,d,c,b,a)
  y= quadgk(@(u)(f4(u,c,b,a).*normcdf(t-u)),d,Inf)-0.05;
  end
```

C.3 Hull & White (2001) method for determining the default thresholds

```
tic
clear all;
t=1:1:5;
p = ones(1,5)*0.05;
deltaT=1;
I=5;
n=5;
M =2000;
K(1)=norminv(p(1))*sqrt(deltaT);
pi=ones(n,M);
x=zeros(n, M);
for m=1:M
    x(1, m)=(5*sqrt(t(1))-K(1))*m/M +K(1);
end

    for i=1:M
             temp =0;
        if(i==M)
              pi(1, i) = 1.0-0.5*erf(-(0.5*(x(1,i)+x(1, i-1))/sqrt(deltaT))/sqrt(2));
        elseif(i==1)
              pi(1, i) = 0.5*erfc(-(0.5*(x(1, i)+x(1,i+1))/sqrt(deltaT))/sqrt(2))-0.5*erfc((-
K(1)/sqrt(deltaT))/sqrt(2));
        else
             pi(1, i)= 0.5*erfc((-0.5*(x(1, i)+x(1,i+1))/sqrt(deltaT))/sqrt(2))-0.5*erfc(-
(0.5*(x(1,i)+x(1, i-1))/sqrt(deltaT))/sqrt(2));
        end
    end

for i=2:I
   m=1:M;
   x(i-1, m)=(5*sqrt(t(i-1))-K(i-1))*m/M +K(i-1);
         for m=1:M
              pi(i-1, m)=  piUpdater2(i,m,pi,x,K,deltaT,M);
         end %for m
        K(i) = fzero(@(k)defaultBcalc(k,i,pi,x,p,deltaT),-2);
end %for i
toc
K


function y=piUpdater2(i,m,pi,x,K,deltaT,M)
% Input the previously determined by the values for the vectors pi,x,K and  scaler p
% This is the vectorised version
 if  i>2
           mm=1:M;
           if(m==1)
              y= sum(pi(i-2, mm).*(normcdf((0.5*(x(i-1,m)+x(i-1,m+1))-x(i-
2,mm))/sqrt(deltaT))-normcdf((K(i-1)-x(i-2, mm))/sqrt(deltaT))));
           elseif (m==M)
              y= sum(pi(i-2, mm).*(1.0-normcdf((0.5*(x(i-1,m)+x(i-1, m-1))-x(i-2,
mm))/sqrt(deltaT))));
           else
             y= sum(pi(i-2, mm).*(normcdf((0.5*(x(i-1,m)+x(i-1,m+1))-x(i-
2,mm))/sqrt(deltaT))-normcdf((0.5*(x(i-1,m)+x(i-1, m-1))-x(i-2, mm))/sqrt(deltaT))));
```

```
        end
 else
    y = pi(1,m);
end
```

C.4 Iscoe & Krenin (unpublished a) Monte Carlo method for the determination of default threshold

```
%---------------------------------------------------------------------------
%This function calculates the default thresholds or boundaries for a given
%vector of assets
% This is used to illustrate the default boundary for a simple cumulative default
% probability term structure of :  Time      CDF %
%                                   1          5
%                                   2         10
%                                   :          :
%                                   5         25
%--------------------------------------------------------------------
format long;
tic
 clear all;
timeStep =1;
Term =5;               % in years
DiscreteTerm= Term*timeStep;
iterations = 1000000
prob = [0:1:4]*0.05;
cprob= 0.05./(1-prob);
 NumberOfAssets=1
   ThresholdA =zeros(1,DiscreteTerm);
      ThresholdA(1,1) = norminv(cprob(1))*sqrt(1/timeStep);
             for k = 2:DiscreteTerm   % k is the time-step -----------------------

                   Storage1 = zeros(1,0);
                    B1=[ ];
                    B1 = randn(iterations,k);
                    B1 = cumsum(B1,2)*sqrt(1/timeStep);
+++ The section determines the threshold values+++++++++++++++++++++++++
                    VecMat1 = [ ];
                    keep1 = [];
                    VecMat1= repmat(ThresholdA(1,1:k-1),iterations,1);
                    keep1 = all(B1(:,1:k-1)>=VecMat1,2);
                    Storage1 = B1(keep1,k);
            %+++ End of section determines the threshold values++++++++++++++++
                   clear B1;
                   clear keep1;
                   clear VecMat1;
                   value= quantile(Storage1,cprob(k));
                   clear Storage1;
                   ThresholdA(1,k)= value;

            end % loop for the main loop (time-step) k ---------------------------------------

ThresholdA
Toc
```

C.5 Morokoff's (2003) method to determine the default thresholds

```
% Purpose:  Calculates the pdf, cdf and inverse cdf of the convolution
% of two distributions, in this case two normals.
%-----------------------------------------------------------------------
% Written by: Renay Singh
% This program implements the algorithm $3.4 of  Kohl et al (2005)
% "General Purpose Algorithm for Distributions in S4-Classes by FFT"
% Requires : consolidator.m  Author : John D'Errico
%----------------------------------------------------------------------

tic
clear;
N=2;
m = 16;   % The default option as given by R code
M = 2^m;
delta=1;
lower =  -10;
upper =  10;
cd=0.05./[1:-0.05:0.8];
db=zeros(1,5);
db(1)= norminv(0.05,0,1);
h = (upper - lower)/M;
y= [lower:h:upper];
p1=  tN(y,db(1),delta);                  % The cdf's of the distns
    for i =2:5

     p2= normcdf(y);
     p1=p1(2:(M+1))-p1(1:M);
     p2=p2(2:(M+1))-p2(1:M);

     pn1= [p1,zeros(1,(N-1)*M)];   % padd with zeros
     fftpn1 = fft(pn1);

     pn2= [p2,zeros(1,(N-1)*M)];   % padd with zeros
     fftpn2 = fft(pn2);

     pn = real(ifft(fftpn1.*fftpn2));
     maxNum= N*M-(N-2);

     pn= [0,pn(1:maxNum)];
     dn = pn/h;
     pn = cumsum(pn);

     x = [N*lower+N/2*h:h:N*upper-N/2*h];
     x = [x(1)-h,x(1),x+h];

     cumDval1=pnfun1(x+0.5*h, x+0.5*h,pn,maxNum);
     tt11=cumDval1/pn(maxNum+1);
     yrange = x+0.5*h;
     [x1,y1]=consolidator(tt11,yrange,'mean');
     db(i)=interp1(x1,y1,cd(i),'linear',0);  % y from above
      p1 =pnfun1(y, x+0.5*h,pn,maxNum);
   p1=tNe(y,p1,cd(i),db(i));
 end
 db;
 toc

function [xcon,ycon,ind] = consolidator(x,y,aggregation_mode,tol)
```

```
% consolidator: consolidate "replicates" in x, also aggregate corresponding y
% usage: [xcon,ycon,ind] = consolidator(x,y,aggregation_mode,tol)
%
% arguments: (input)
%  x - rectangular array of data to be consolidated. If multiple (p)
%      columns, then each row of x is interpreted as a single point in a
%      p-dimensional space. (x may be character, in which case xcon will
%      be returned as a character array.)
%
%      x CANNOT be complex. If you do have complex data, split it into
%      real and imaginary components as columns of an array.
%
%      If x and y are both ROW vctors, they will be transposed and
%      treated as single columns.
%
%  y - outputs to be aggregated. If y is not supplied (is left empty)
%      then consolidator is similar to unique(x,'rows'), but with a
%      tolerance on which points are distinct. (y may be complex.)
%
%      y MUST have the same number of rows as x unless y is empty.
%
%  aggregation_mode - (OPTIONAL) - an aggregation function, either
%      in the form of a function name, such as 'mean' or 'sum', or as
%      a function handle, i.e., @mean or @std. An inline function would
%      also work, for those users of older matlab releases.
%
%      DEFAULT: 'mean'
%
%      Aggregation_mode may also be the string 'count', in which case
%      a count is made of the replicates found. Ycon will only have
%      one column in this case.
%
%      The function supplied MUST have the property that it operates
%      on the the first dimension of its input by default.
%
%      Common functions one might use here are:
%      'mean', 'sum', 'median', 'min', 'max', 'std', 'var', 'prod'
%      'geomean', 'harmmean'.
%
%      These last two examples would utilize the statistics toolbox,
%      however, these means can be generated using a function
%      handle easily enough if that toolbox is not available:
%
%      fun = @(x) 1./mean(1./x)        % harmonic mean
%      fun = @(x) exp(mean(log(x)))    % geometric mean
%
%  tol - (OPTIONAL) tolerance to identify replicate elements of x. If
%      x has multiple columns, then the same (absolute) tolerance is
%      applied to all columns of x.
%
%      DEFAULT: 0
%
% arguments: (output)
%  xcon - consolidated x. Replicates wthin the tolerance are removed.
%      if no y was specified, then consolidation is still done on x.
%
%  ycon - aggregated value as specified by the aggregation_mode.
%
%  ind  - vector - denotes the elements of the original array which
%      were consolidated into each element of the result.
```

```
%
%
% Example 1:
%
% Group means: (using a function handle for the aggregation)
%  x = round(rand(1000,1)*5);
%  y = x+randn(size(x));
%  [xg,yg] = consolidator(x,y,@mean);
%  [xg,yg]
%  ans =
%         0    0.1668
%    1.0000    0.9678
%    2.0000    2.0829
%    3.0000    2.9688
%    4.0000    4.0491
%    5.0000    4.8852
%
% Example 2:
%
% Group counts on x
%  x = round(randn(100000,1));
%  [xg,c] = consolidator(x,[],'count');
%  [xg,c]
%  ans =
%        -4          26
%        -3         633
%        -2        5926
%        -1       24391
%         0       38306
%         1       24156
%         2        5982
%         3         559
%         4          21
%
% Example 3:
%
% Unique(x,'rows'), but with a tolerance
%  x = rand(100,2);
%  xc = consolidator(x,[],[],.05);
%  size(xc)
%  ans =
%      62     2
%
% See also: unique
%
% Author: John D'Errico
% e-mail address: woodchips@rochester.rr.com
% Release: 3
% Release date: 5/2/06

% is it a character array?
if ischar(x)
  charflag = 1;
  x=double(x);
else
  charflag = 0;
end

% check for/supply defaults
if (nargin<4) || isempty(tol)
```

```
  tol = 0;
end
if (tol<0)
  error 'Tolerance must be non-negative.'
end
tol = tol*(1+10*eps);

% -------------------------------------------------------
% DETERMINE AGGREGATION MODE AND CREATE A FUNCTION HANDLE
% -------------------------------------------------------
if (nargin < 3) || isempty(aggregation_mode)
    % use default function
  fun=@mean;
  aggregation_mode='mean';

elseif ischar(aggregation_mode)
    aggregation_mode=lower(aggregation_mode);

  k=strmatch(aggregation_mode,'count');
  if ~isempty(k)
    fun=@(x) x;
    else
        fun=str2func(aggregation_mode);
  end

elseif isa(aggregation_mode,'inline')
  fun=aggregation_mode;
  am = struct(fun);
  aggregation_mode=am.expr;

else
  fun=aggregation_mode;
  aggregation_mode=func2str(fun);

end
% -------------------------------------------------------

% was y supplied, or empty?
[n,p] = size(x);
if (nargin<2) || isempty(y)
  y = zeros(n,0);
  fun = @(x) x;
  aggregation_mode = 'count';
end
% check for mismatch between x and y
[junk,q] = size(y);
if n~=junk
  error 'y must have the same number of rows as x.'
end

% are both x and y row vectors?
if (n == 1)
  x=x';
  n = length(x);
  p = 1;

  if ~isempty(y)
    y=y';
  else
    y=zeros(n,0);
  end

  q = size(y,2);
end

if isempty(y)
  aggregation_mode = 'count';
end

% consolidate elements of x.
% first shift, scale, and then ceil.
if tol>0
  xhat = x - repmat(min(x,[],1),n,1)+tol*eps;
  xhat = ceil(xhat/tol);
else
  xhat = x;
end
[xhat,tags] = sortrows(xhat);
x=x(tags,:);
y=y(tags,:);

% count the replicates
iu = [true;any(diff(xhat),2)];
eb = cumsum(iu);

% which original elements went where?
if nargout>2
  ind = eb;
  ind(tags) = ind;
end

% count is the vector of counts for the consolidated
% x values
if issparse(eb)
  eb = full(eb);
end
count=accumarray(eb,1).';
% ec is the expanded counts, i.e., counts for the
% unconsolidated x
ec = count(eb);

% special case for aggregation_mode of 'count',
% but we still need to aggregate (using the mean) on x
if strcmp(aggregation_mode,'count')
  ycon = count.';
  q = 0;   % turn off aggregation on y
else
  ycon = zeros(length(count),q);
end

% loop over the different replicate counts, aggregate x and y
ucount = unique(count);
xcon = repmat(NaN,[length(count),p]);
fullx = ~issparse(x);
fully = ~issparse(y);
for k=ucount
  if k==1
    xcon(count==1,:) = x(ec==1,:);
  else
    if fullx
```

```
      v=permute(x(ec==k,:),[3 2 1]);
    else
      v=permute(full(x(ec==k,:)),[3 2 1]);
    end
    v=reshape(v,p,k,[]);
    v=permute(v,[2 1 3]);
    xcon(count==k,:)=reshape(mean(v),p,[]).';
  end

  if q>0
    % aggregate y as specified
    if k==1
      switch aggregation_mode
        case {'std' 'var'}
          ycon(count==1,:) = 0;
        otherwise
          ycon(count==1,:) = y(ec==1,:);
      end
    else
      if fully
        v=permute(y(ec==k,:),[3 2 1]);
      else
        v=permute(full(y(ec==k,:)),[3 2 1]);
      end
      v=reshape(v,q,k,[]);
      v=permute(v,[2 1 3]);

      % aggregate using the appropriate function
      ycon(count==k,:)=reshape(fun(v),q,[]).';

    end
  end
end

% was it originally a character array?
if charflag
  xcon=char(xcon);
end
```

C.6 Program to determine the CDO tranche spreads using the multistep approach

```
tic
clear all;
fprintf('____________________________________________________________\n')
fprintf('____________________________________________________________\n')
format short g
format compact
load Table59.mat
load ThresholdA.mat
RiskFree = 0.05;
corr = 0.5;
fprintf('Correlation: %g.\r',corr)
Rec = 0.3;
fprintf('Recovery: %g.\r',Rec)
Delta = 0.25;
Term =5;                % in years
timeStep =4;          % fractions of a year 52=weekly, 365 = daily
Tm= Term/Delta;    % Number of premium payments
Portfolio =[Table59(:,2)*1*(1-Rec),Table59(:,4)/10000];
hazardRate= Portfolio(:,2)'/(1-Rec); % convert to harzard rate.
NumberOfAssets=size(hazardRate,2);       % size of the basket
DiscreteTerm= Term*timeStep;  % timeStep =1; % fractions of a year 52=weekly, 365 =
daily

% Generate the correlation matrix;
 R = repmat(corr,NumberOfAssets,NumberOfAssets)+(1-corr)*eye(NumberOfAssets);
 v = chol(R);

UpperTranche = 400
LowerTranche = 150
TrancheSize = UpperTranche-LowerTranche;
TotalPremiumsE = 0;
TotalBenefitsE=  0;

iterations = 100000
 Threshold =ThresholdA;

for i =1:iterations  % Main Simulation
loop=============================================
    % Generate a 50x20 matrix of correlated multivariate random numbers...
        z = randn(NumberOfAssets,DiscreteTerm);
        A = z'*v;
        A= cumsum(A',2)*sqrt(1/timeStep);
       Defaults= zeros(NumberOfAssets,2);
               for Entity = 1:NumberOfAssets
%*****************************************************
                   for k =1: DiscreteTerm
                         if Defaults(Entity,1) == 0 % i.e. no default then test for default
                                   if   A(Entity,k) <= Threshold(Entity,k) % if the asset(entity) defaults
                                      Defaults(Entity,1) =1; % set the asset to default state
                                      Defaults(Entity,2) = k ;  % store the default time
                                  end
                         end     % of default state test [If statement]---------------------------------
                end   %****************** End of k updates***********************
         end % for entity loop
```

```
% ---------------------------------------------------------------------------------------------------
------------------
% Obtain the defaulted assets and sort the times of default keeping track of the asset values
      Defaults = [Defaults,Portfolio(:,1)];
      DefaultIndex=find(Defaults(:,1)==1);
      DefaultedAssets = Defaults(DefaultIndex,3);
      DefaultTime1 = Defaults(DefaultIndex,2);
      [DefaultTime1,AssetDefOrder] = sort(Defaults(DefaultIndex,2));
      DefaultedAssets = DefaultedAssets(AssetDefOrder);  % change to order of default; order
must be the same as Defaulted Assets
      numDefTimes1 = size(DefaultedAssets',2) ;        % Apply convention; all these vectors
are row vectors
      DefaultTime1 = DefaultTime1*(1/timeStep);   % Back to reality, qrtl'y periods
   % ------------------------------------------------------------------------
  BenefitPV_E1 =0;
  PremiumPV_E1=0;

 if isempty(DefaultTime1) ==1
        BenefitPV_E1=0;        % No change here
        PortfolioLoss_t = zeros(1,NumberOfAssets);
        TotalLoss = zeros(1,NumberOfAssets);
        PremiumPV_E1 = Delta*sum(exp(-RiskFree*(Delta:Delta:Term)))*TrancheSize;

else
        PortfolioLoss_t= DefaultedAssets';
        TotalLoss = cumsum(PortfolioLoss_t);

          if    (LowerTranche <=  TotalLoss(end))  && (TotalLoss(end)< UpperTranche)

              minTime=0;
              minTime = min(find(TotalLoss>=LowerTranche));
              BenefitPV_E1 = (TotalLoss(minTime)-LowerTranche)*exp(-
RiskFree*DefaultTime1(minTime));
                   for t = minTime+1:numDefTimes1
                      BenefitPV_E1 = PortfolioLoss_t(t)*exp(-RiskFree*DefaultTime1(t)) +
BenefitPV_E1;
                   end
             elseif  TotalLoss(end) >= UpperTranche   %------------------------------------------
--------
               minTime=0;
               maxTime=0;
               maxTime = min(find(TotalLoss>=UpperTranche));
               minTime = min(find(TotalLoss>=LowerTranche));
               BenefitPV_E1 = min(TotalLoss(minTime)-LowerTranche,TrancheSize)*exp(-
RiskFree*DefaultTime1(minTime));
                    if  minTime+1 <=  maxTime-1
                      for t = minTime+1:maxTime-1
                        BenefitPV_E1 = PortfolioLoss_t(t)*exp(-RiskFree*DefaultTime1(t)) +
BenefitPV_E1;
                      end
                    end
               if minTime ~=maxTime
                 BenefitPV_E1 = max(UpperTranche-(TotalLoss(maxTime)-
PortfolioLoss_t(maxTime)),0)*exp(-RiskFree*DefaultTime1(maxTime))+BenefitPV_E1;
               end %......... if
           end % ............if
 %-------------This loop determines the PV of Premium payments   -------------------------------
       PortfolioLoss_t1=zeros(1,Tm);
          for k = 1:Tm
            PortfolioLoss_t1(k) = 0;
```

```
                for i = 1:numDefTimes1
                     if  DefaultTime1(i) <= k*Delta
                        PortfolioLoss_t1(k) = DefaultedAssets(i)+ PortfolioLoss_t1(k);
                     end
                 end
          end
         PremiumPV_E1 = 0;
         for k = 1:Tm
           PremiumPV_E1 = Delta*exp(-RiskFree*(k*Delta))*(min(max(UpperTranche-
PortfolioLoss_t1(k),0), TrancheSize))+PremiumPV_E1;
         end

%---------------------------------END OF PREMIUM LEG
 end
 %----------------BENEFIT LEG ----
*******************************************************

   TotalPremiumsE = TotalPremiumsE + PremiumPV_E1;
   TotalBenefitsE = TotalBenefitsE + BenefitPV_E1;
end   %  End of iterations loop------------------------------------------------------------------
------
TrancheE=(TotalBenefitsE/TotalPremiumsE)*10000;
fprintf('Spread in bps: %g.\r',TrancheE)
fprintf('\r')

toc
```

C.7 Program to determine the CDO tranche spreads using the default time approach

```
function [TrancheE, eTime] =
fairTrancheCh6_Li(corr,iterations,LowerTranche,UpperTranche,Table59)
%----------------------------------------------------
tic
TrancheSize = UpperTranche-LowerTranche;
Term =5;
Delta =  0.25;          % Frequency of Premium payment p.a.
Tm = Term/Delta;      % This is the total number of time steps for the premium leg.
 RiskFree= 0.05;


 Assets = Table59(:,2);       % Note that these are recovery adjusted
 hazard =Table59(:,4);             % expressed as basis percentage

 NumberOfAssets=size(Assets,1);
  reciprocal_hazardRate =ones(NumberOfAssets,1)./hazard;

TotalPremiumsE = 0;
TotalBenefitsE=  0;

v=[1,repmat(corr,1,NumberOfAssets-1)];
R=toeplitz(v);
A = chol(R);

for Sims =1:iterations   % START OF SIMULATION*********************

    DefaultTime1 =[ ];
    PortfolioLoss_t = [ ];
    TotalLoss =[];
    DefaultVector=[];
    AssetDefOrder=[];

    z = randn(NumberOfAssets,1);
    u = z'*A;
    u = normcdf(u,0,1);

      DefaultTime1 =((-reciprocal_hazardRate').*log(1-u))  ;
      DefaultVector = find(DefaultTime1<=Term) ;

      DefaultedAssets=Assets(DefaultVector);

     [DefaultTime1,AssetDefOrder]=sort(DefaultTime1(DefaultVector)) ;

      DefaultedAssets = DefaultedAssets(AssetDefOrder);  % change to order of default

    numDefTimes1 = size(DefaultedAssets',2);          % Apply convention; all these vectors are
row vectors
    % -----------------------------------------------------------------------
           if isempty(DefaultTime1) ==1
                  PortfolioLoss_t = zeros(1,NumberOfAssets);
                  TotalLoss = zeros(1,NumberOfAssets);
            else
                  PortfolioLoss_t= DefaultedAssets';
                  TotalLoss = cumsum(PortfolioLoss_t);
```

```
       end
 %-----------------BENEFIT LEG --------------------------------------------
 BenefitPV_E1=0;

 if  LowerTranche>=0
      if TotalLoss(end)==0
         BenefitPV_E1 = 0;

       elseif    (LowerTranche <  TotalLoss(end))  && (TotalLoss(end)< UpperTranche)

          minTime=0;
          minTime = min(find(TotalLoss>=LowerTranche));
          BenefitPV_E1 = 0;

               BenefitPV_E1 = (TotalLoss(minTime)-LowerTranche)*exp(-
RiskFree*DefaultTime1(minTime));

                for t = minTime+1:numDefTimes1
                  BenefitPV_E1 = PortfolioLoss_t(t)*exp(-RiskFree*DefaultTime1(t)) +
BenefitPV_E1;
                end

      elseif   TotalLoss(end) >= UpperTranche   %------------------------------------------------
---
             minTime=0;
             maxTime=0;
             BenefitPV_E1=0;

             maxTime = min(find(TotalLoss>=UpperTranche));
             minTime = min(find(TotalLoss>=LowerTranche));
             BenefitPV_E1 = min(TotalLoss(minTime)-LowerTranche,TrancheSize)*exp(-
RiskFree*DefaultTime1(minTime));

            if  minTime+1 <=  maxTime-1
                  for t = minTime+1:maxTime-1
                   BenefitPV_E1 = PortfolioLoss_t(t)*exp(-RiskFree*DefaultTime1(t)) +
BenefitPV_E1;
                  end
            end
            if minTime ~=maxTime
              BenefitPV_E1 = max(UpperTranche-(TotalLoss(maxTime)-
PortfolioLoss_t(maxTime)),0)*exp(-RiskFree*DefaultTime1(maxTime))+BenefitPV_E1 ;


            end
    end
 end

 %---------------------- determines the PV of Premium payments   ---------------------
PortfolioLoss_t1=[];
PortfolioLoss_t1=zeros(1,Tm);
          for k = 1:Tm
         PortfolioLoss_t1(k) = 0;

             for i = 1:numDefTimes1
                 if  DefaultTime1(i) <= k*Delta
                   PortfolioLoss_t1(k) = DefaultedAssets(i)+ PortfolioLoss_t1(k);
                end
         end
        end
```

```
            PremiumPV_E1 = 0;

          for k = 1:Tm
            PremiumPV_E1 = Delta*exp(-RiskFree*(k*Delta))*(min(max(UpperTranche-
PortfolioLoss_t1(k),0), TrancheSize))+PremiumPV_E1;
          end

%-----------------------------------END OF PREMIUM LEG

    TotalPremiumsE = TotalPremiumsE + PremiumPV_E1;
    TotalBenefitsE = TotalBenefitsE + BenefitPV_E1;
end
TrancheE=(TotalBenefitsE/TotalPremiumsE)*10000;
eTime=toc;
```

C.8 Exact recursion method to determine the credit loss distribution and the value of CDO tranche spreads

```
clear all
tic
format long
load Table59.mat
Assets =[Table59(:,2)*10,Table59(:,4)/10000];
rho =sqrt(0.5);
Rec =0.3;
RiskFree = 0.05;
h = Assets(:,2)/(1-Rec);
GaussPoints =20

[abscissas,weights]=qrule(GaussPoints,9,1,1);
abscissas=sqrt(2)*(abscissas);    % adjust to Normal Distribution...
weights=(weights)/sqrt(pi);         %  ditto

MaxLoss =80;
Term =5;
delta =0.25;
ValSteps =Term/delta;
RecAssets=Assets(:,1)*(1-Rec);
GreatCommonDivisor= gcd2(RecAssets)
% Error = 5;
% LossUnit = ASB_LossUnit(RecAssets,Error);

RecAssets=RecAssets/GreatCommonDivisor;

% conditionalLoss=zeros(MaxLoss+1,GaussPoints);
unconditionalLoss=zeros(MaxLoss+1,ValSteps);

for t =delta:delta:Term
   conditionalLoss=zeros(MaxLoss+1,GaussPoints);
   prob = zeros(50,20);
   prob= p_VectorM(t,rho,abscissas,h);

  for k= 1:GaussPoints
%     V =  abscissas(k);
      A= zeros(MaxLoss+1,50);
      %------------------------------------------------------------------------
      % NOTE: A is defined as a (1 xMaxLoss+1) matrix.
      % to store values each loss must be incremented by one as Matlab does not
      % allow zero based indexing
      %------------------------------------------------------------------------


      % Start by setting up the first colomn of the conditional loss matrix

      A(1,1) = (1-prob(1,k));
      A(2: RecAssets(1),1) = 0;
      A( RecAssets(1)+1,1)= prob(1,k);
      A( RecAssets(1)+2:MaxLoss+1,1)=0;

      for K= 2:50
```

```
        loss = 0:RecAssets(K)-1;
              A(loss+1, K) = A(loss+1,K-1)*(1-prob(K,k));
        loss = RecAssets(K):MaxLoss;
              A(loss+1, K) = A(loss+1,K-1)*(1-prob(K,k)) + A(loss-RecAssets(K)+1,K-
1)*prob(K,k);

       end  % for loop ==> K

        conditionalLoss(:,k) = A(:,end);  % For each time Gauss-Hermite point, collect one
vector

      end % loop for Gaussian Quadrature

   adjustedWeights =  repmat(weights',MaxLoss+1,1);
   unconditionalLoss(:, t/delta) = sum(conditionalLoss.*adjustedWeights,2);

end %  end of time loop
%------------------------------------------------------------------------

Lower =0;
Upper =250;
lossVector =0:1:MaxLoss;
time = delta:delta:Term;
ExpectedLossTime = sum(unconditionalLoss.*repmat(TrancheLoss(Lower,Upper,
GreatCommonDivisor*lossVector'),1,ValSteps),1);
ExpectedLossTime_lessOne = ExpectedLossTime;
ExpectedLossTime_lessOne(end) =0; % make last  element zero
ExpectedLossTime_lessOne=ExpectedLossTime_lessOne([ end 1:end-1 ]); % shift right/down
1 element

DiffExpectedLossTime =diff([0,ExpectedLossTime]);

timeDefaultLeg = (delta/2):delta:Term;   % 0.5*(T(i+1)+T(i)) ASB page 69

DefaultLeg = sum(exp(-RiskFree*timeDefaultLeg).*DiffExpectedLossTime);
PremiumLeg =delta*sum(exp(-RiskFree*time).*(Upper-Lower -
0.5*(ExpectedLossTime+ExpectedLossTime_lessOne)));
%-----------------------------------------------------------------------
100*DefaultLeg/PremiumLeg

% fprintf('Spread in %% terms: %g.\r',100*DefaultLeg/PremiumLeg)
% fprintf('.....................................................................\n')
% disp(datestr(now))
toc


function [bp,wf]=qrule(n,wfun,alpha,beta)
%QRULE computes abscissas and weight  factors for some selected
%   Gaussian quadratures.
%
%CALL:  [bp,wf]=qrule(n,wfun,alpha,beta)
%
%  bp = base points
%  wf = weight factors
%  n  = number of base points (abscissas) (integrates a (2n-1)th order
%       polynomial exactly)
%wfun = weight function%
%     1  p(x)=1                       a =-1,   b = 1 Legendre (default)
%     2  p(x)=1/sqrt((x-a)*(b-x)),    a =-1,   b = 1 Chebyshev of the
%                                                  first kind
%     3  p(x)=sqrt((x-a)*(b-x)),      a =-1,   b = 1 Chebyshev of the
%                                                 second kind
%     4  p(x)=sqrt((x-a)/(b-x)),      a = 0,   b = 1
%     5  p(x)=1/sqrt(b-x),            a = 0,   b = 1
%     6  p(x)=sqrt(b-x),              a = 0,   b = 1
%     7  p(x)=(x-a)^alpha*(b-x)^beta  a =-1,   b = 1 Jacobi
%                              alpha, beta >-1 (default alpha=beta=0)
%     8  p(x)=x^alpha*exp(-x)         a = 0,   b = inf generalized Laguerre
%     9  p(x)=exp(-x^2)               a =-inf, b = inf Hermite
%    10  p(x)=1                       a =-1,   b = 1 Legendre (slower than 1)
%
%  The Gaussian Quadrature integrates a (2n-1)th order
%  polynomial exactly and the integral is of the form
%           b                         n
%          Int ( p(x)* F(x) ) dx  =  Sum ( wf_j* F( bp_j ) )
%           a                        j=1
%  See also: gaussq


% Reference
%   wfun 1: copied from grule.m in NIT toolbox, see ref [2]
%   wfun 2-6: see ref [4]
%   wfun 7-10:  Adapted from Netlib routine gaussq.f see ref [1,3]
%
% [1]  Golub, G. H. and Welsch, J. H. (1969)
% 'Calculation of Gaussian Quadrature Rules'
%  Mathematics of Computation, vol 23,page 221-230,
%
% [2] Davis and Rabinowitz (1975) 'Methods of Numerical Integration', page 365,
%     Academic Press.
%
% [3]. Stroud and Secrest (1966), 'gaussian quadrature formulas',
%      prentice-hall, Englewood cliffs, n.j.
%
% [4] Abromowitz and Stegun (1954) ''


%  By Bryce Gardner, Purdue University, Spring 1993.
% Modified by Per A. Brodtkorb 19.02.99 pab@marin.ntnu.no
% to compute other quadratures  than the default
if nargin<4|isempty(beta),
 beta=0;
end


if nargin<3|isempty(alpha),
  alpha=0;
end
if alpha<=-1 | beta <=-1,
  error('alpha and beta must be greater than -1')
end


if nargin<2|isempty(wfun),
  wfun=1;
end
```

```
switch wfun, %
 case 1,
  %  This routine computes Gauss base points and weight factors
  %  using the algorithm given by Davis and Rabinowitz in 'Methods
  %  of Numerical Integration', page 365, Academic Press, 1975.
  bp=zeros(n,1); wf=bp; iter=2; m=fix((n+1)/2); e1=n*(n+1);
  mm=4*m-1; t=(pi/(4*n+2))*(3:4:mm); nn=(1-(1-1/n)/(8*n*n));
  xo=nn*cos(t);
  for j=1:iter
   pkm1=1; pk=xo;
   for k=2:n
    t1=xo.*pk; pkp1=t1-pkm1-(t1-pkm1)/k+t1;
    pkm1=pk; pk=pkp1;
   end
   den=1.-xo.*xo; d1=n*(pkm1-xo.*pk); dpn=d1./den;
   d2pn=(2.*xo.*dpn-e1.*pk)./den;
   d3pn=(4*xo.*d2pn+(2-e1).*dpn)./den;
   d4pn=(6*xo.*d3pn+(6-e1).*d2pn)./den;
   u=pk./dpn; v=d2pn./dpn;
   h=-u.*(1+(.5*u).*(v+u.*(v.*v-u.*d3pn./(3*dpn))));
   p=pk+h.*(dpn+(.5*h).*(d2pn+(h/3).*(d3pn+.25*h.*d4pn)));
   dp=dpn+h.*(d2pn+(.5*h).*(d3pn+h.*d4pn/3));
   h=h-p./dp; xo=xo+h;
  end
  bp=-xo-h;
  fx=d1-h.*e1.*(pk+(h/2).*(dpn+(h/3).*(...
    d2pn+(h/4).*(d3pn+(.2*h).*d4pn))));
  wf=2*(1-bp.^2)./(fx.*fx);
  if (m+m) > n, bp(m)=0; end
  if ~((m+m) == n), m=m-1; end
  jj=1:m; n1j=(n+1-jj); bp(n1j)=-bp(jj); wf(n1j)=wf(jj);
  % end

case 2, % p(x)=1/sqrt((x-a)*(b-x)), a=-1 and b=1 (default)
 j=[1:n];
 wf = ones(1,n) * pi / n;
 bp=cos( (2*j-1)*pi / (2*n) );


case 3, %p(x)=sqrt((x-a)*(b-x)),   a=-1   and b=1
 j=[1:n];
 wf = pi/ (n+1) *sin( j*pi / (n+1) ).^2;
 bp=cos( j*pi / (n+1) );


case 4, %p(x)=sqrt((x-a)/(b-x)),   a=0   and b=1
  j=[1:n];
  bp=cos( (2*j-1)*pi /2/ (2*n+1) ).^2;
  wf=2*pi.*bp/(2*n+1) ;


case 5, % %p(x)=1/sqrt(b-x),   a=0   and b=1
  [bp wf]=grule(2*n);
 wf(bp<0)=[];
 wf=wf*2;
  bp(bp<0)=[];
 bp=1-bp.^2;


case 6, % %p(x)=sqrt(b-x),   a=0   and b=1
```

```
  [bp wf]=grule(2*n+1);
 wf(bp<=0)=[];
  bp(bp<=0)=[];
 wf=2*bp.^2.*wf;
 bp=1-bp.^2;

case {7,8,9,10} ,%
 %7 p(x)=(x-a)^alpha*(b-x)^beta a=-1 b=1 Jacobi
 %8 p(x)=x^alpha*exp(-x) a=0,   b=inf generalized Laguerre
 %9 p(x)=exp(-x^2)       a=-inf, b=inf Hermite
 %10 p(x)=1              a=-1 b=1        Legendre slower than 1
 % this procedure uses the coefficients a(j), b(j) of the
 %      recurrence relation
 %
 %          b p (x) = (x - a ) p   (x) - b   p   (x)
 %           j j           j   j-1       j-1 j-2
 %
 %      for the various classical (normalized) orthogonal polynomials,
 %      and the zero-th moment
 %
 %          muzero = integral w(x) dx
 %
 %      of the given polynomial's weight function w(x).  since the
 %      polynomials are orthonormalized, the tridiagonal matrix is
 %      guaranteed to be symmetric.
 %
 %
 %         the input parameter alpha is used only for laguerre and
 %      jacobi polynomials, and the parameter beta is used only for
 %      jacobi polynomials.  the laguerre and jacobi polynomials
 %      require the gamma function.


 a=zeros(n,1);
 b=zeros(n-1,1);
 switch wfun
   case 7,  %jacobi
     ab = alpha + beta;
     abi = 2 + ab;
     muzero = 2^(ab + 1) * gamma(alpha + 1) * gamma(beta + 1) / gamma(abi);
     a(1) = (beta - alpha)/abi;
     b(1) = sqrt(4*(1 + alpha)*(1 + beta)/((abi + 1)*abi^2));
     a2b2 = beta^2 - alpha^2;

     i = (2:n-1)';
     abi = 2*i + ab;
     a(i) = a2b2./((abi - 2).*abi);
     a(n) =a2b2./((2*n - 2+ab).*(2*n+ab));
     b(i) = sqrt (4*i.*(i + alpha).*(i + beta)*(i + ab)./((abi.^2 - 1).*abi.^2));

   case 8, % Laguerre
     muzero=gamma(alpha+1);
     i = (1:n-1)';
     a(i) = 2 .* i - 1 + alpha;
     a(n)=2*n-1+alpha;
     b = sqrt( i .* (i + alpha) );
   case 9, %Hermite
     i = (1:(n-1))';
     muzero = sqrt(pi);
     %a=zeros(m,1);
```

```
      b=sqrt(i/2);
    case 10,  % legendre NB! much slower than wfun=1
      muzero = 2;
      i = (1:n-1)';
      abi = i;
      b(i) = abi./sqrt(4*abi.^2 - 1);

  end

  %[v d] = eig( full(spdiags([b a b],-1:1,n,n )));
  [v d] = eig( diag(a) + diag(b,1) + diag(b,-1) );
  wf = v(1,:);
  if 1,
    [bp i] = sort( diag(d) );
    wf = wf(i);
  else % save some valuable time by not sorting
    bp = diag(d) ;
  end
  bp=bp;

  wf = muzero.* wf.^2;
  wf =wf';

otherwise, error('unknown weight function')
end
```

C.9 Numerical inversion of the characteristic function method to determine the credit loss distribution

```
%-------------------------------------------------------------------
% This program evaluates the discrete credit loss distribution
% via the inversion of the CF.
%
%This program calls the following sub-programs
%
% qrule.m :   This determines the abscissas and weights for the Guassian
%             quadrature.
%
% CharacteristicFunction8.m : Determines the characteristic function of the
%                             portfolio. This function needs
%                             p_VectorGH
%
% p_VectorGH.m : Determines the conditional default probability (given a
%                particular state of the economy.
%
%-------------------------------------------------------------------

clear all
tic
load Table59.mat

Assets =[Table59(:,1)*10,Table59(:,2)/10000];
rho=sqrt(0.5);
Rec =0.3;
RiskFree = 0.05;
delta =0.25;
Term =5;
ValSteps =Term/delta;

Total= sum(Assets(:,1)*(1-Rec))
h = Assets(:,2)'/(1-Rec);
M=Assets(:,1)'*(1-Rec)/Total;
pmin=7;
ResF= 2^pmin

%  -----------------------------/--------
[bp,wf]=qrule(20,9,1,1);
abscissa=sqrt(2)*(bp);
weights=(wf)/sqrt(pi);
% -------------------------------------
dx =0.0125  % GreatCommonDivisor
t= 2*pi*[0:1:ResF-1]/(dx*ResF);
paymentDates=[0.25:0.25:5];
for paymentNumber=1:20
   FourierT=[];
   for j=1:ResF/2+1
     FourierT(j) =
CharacteristicFunction8(t(j),paymentDates(paymentNumber),weights,abscissa,h,rho,M);
   end
  FourierT(ResF/2+2:ResF) =fliplr(conj(FourierT(2:ResF/2)));
 unconditionalLoss(paymentNumber,:)=ifft(FourierT);
end
```

```
MaxLoss =80;
unconditionalLoss=real(unconditionalLoss');
unconditionalLoss =unconditionalLoss(1:(MaxLoss+1),1:ValSteps);

dx=Total*dx;

Lower =1500;
Upper =4000;
lossVector =0:1:MaxLoss;
time = delta:delta:Term;
ExpectedLossTime = sum(unconditionalLoss.*repmat(TrancheLoss(Lower,Upper,
dx*lossVector'),1,ValSteps),1);
ExpectedLossTime_lessOne = ExpectedLossTime;
ExpectedLossTime_lessOne(end) =0; %make last  element zero
ExpectedLossTime_lessOne=ExpectedLossTime_lessOne([ end 1:end-1 ]); % shift right/down
1 element

DiffExpectedLossTime =diff([0,ExpectedLossTime]);

timeDefaultLeg = (delta/2):delta:Term;   % 0.5*(T(i+1)+T(i)) ASB page 69

DefaultLeg = sum(exp(-RiskFree*timeDefaultLeg).*DiffExpectedLossTime);
PremiumLeg =delta*sum(exp(-RiskFree*time).*(Upper-Lower -
0.5*(ExpectedLossTime+ExpectedLossTime_lessOne)));
100*DefaultLeg/PremiumLeg
toc

function outx =CharacteristicFunction8(t,time,weights,abscissa,h,rho,M)
 sizeGH= size(weights,1);

 outx= sum(prod((1+ (p_VectorGH(time,rho,abscissa,h)).*repmat(exp(-M*t*i)-
1,sizeGH,1)),2).*weights);


function yy=p_VectorGH(t,rho,V,h)

yy=normcdf((repmat(norminv(1-exp(-h*t)),size(V,1),1)-rho*repmat(V,1,size(h,2)))/(sqrt(1-
rho^2)),0,1);
```

C.10 Compound Poisson approximation using Panjer recursion to determine the credit loss distribution

```
clear all
tic
format long
 load Table59.mat
Assets =[Table59(:,2)*10,Table59(:,4)/10000];

rho =sqrt(0.5);

Rec =0.3;
RiskFree = 0.05;
h = Assets(:,2)/(1-Rec);
GaussPoints =20

[abscissas,weights]=qrule(GaussPoints,9,1,1);
abscissas=sqrt(2)*(abscissas);     % adjust to Normal Distribution...
weights=(weights)/sqrt(pi);           %  ditto

MaxLoss =80;
Term =5;
delta =0.25;
ValSteps =Term/delta;
RecAssets=Assets(:,1)*(1-Rec);
GreatCommonDivisor= gcd2(RecAssets)
RecAssets=RecAssets/GreatCommonDivisor;

AssetTemp =repmat(RecAssets,1,max(RecAssets));
AssetRange = repmat(min(RecAssets):max(RecAssets),size(RecAssets),1);
result = (AssetTemp==AssetRange)';

% conditionalLoss=zeros(MaxLoss+1,GaussPoints);
unconditionalLoss=zeros(MaxLoss+1,ValSteps);

for t =delta:delta:Term
   conditionalLoss=zeros(MaxLoss+1,GaussPoints);

  for k= 1:GaussPoints
     V =  abscissas(k);
      g = zeros(MaxLoss+1,1);

   % Calculate the probabilities and store the vector.
   prob = zeros(1,50);
   prob=  p_Vector(t,rho,V,h)';

   g(1)=exp(-sum(prob));    % This is the value of g(0) as per Panjer.
   % Best to calculate g(0) now before var prob is rearranged.
   probloss= repmat(prob',1,3).*result';
   probloss=[probloss,zeros(50,78)];

   for loss=1:MaxLoss
      for xx=1:loss
         g(loss+1) = xx*sum(probloss(:,xx))*g(loss-xx+1)/loss+g(loss+1);
      end
   end
   conditionalLoss(:,k) = g;  % For each time Gauss-Hermite point, collect the vector
```

```
    end % loop for Gaussian Quadrature
   adjustedWeights =  repmat(weights',MaxLoss+1,1);
   unconditionalLoss(:, t/delta) = sum(conditionalLoss.*adjustedWeights,2);

end %  end of time loop
%-------------------------------------------------------------------------
Lower =750;
Upper =1500;
lossVector =0:1:MaxLoss;
time = delta:delta:Term;
ExpectedLossTime = sum(unconditionalLoss.*repmat(TrancheLoss(Lower,Upper,
GreatCommonDivisor*lossVector'),1,ValSteps),1);
ExpectedLossTime_lessOne = ExpectedLossTime;
ExpectedLossTime_lessOne(end) =0; %make last  element zero
ExpectedLossTime_lessOne=ExpectedLossTime_lessOne([ end 1:end-1 ]); % shift right/down
1 element

DiffExpectedLossTime =diff([0,ExpectedLossTime]);

timeDefaultLeg = (delta/2):delta:Term;   % 0.5*(T(i+1)+T(i)) ASB page 69

DefaultLeg = sum(exp(-RiskFree*timeDefaultLeg).*DiffExpectedLossTime);
PremiumLeg =delta*sum(exp(-RiskFree*time).*(Upper-Lower -
0.5*(ExpectedLossTime+ExpectedLossTime_lessOne)));

100*DefaultLeg/PremiumLeg
toc
```

C.11 Compound Poisson approximation implementation using the FFT to determine the value of CDO tranche spreads

```
clear all
tic
load Table59.mat
Assets =[Table59(:,2)*10,Table59(:,4)/10000];
rho =sqrt(0.5);
Rec =0.3;
RiskFree = 0.05;
h = Assets(:,2)/(1-Rec);
GaussPoints =20

[abscissas,weights]=qrule(GaussPoints,9,1,1);
abscissas=sqrt(2)*(abscissas);     % adjust to Normal Distribution...
weights=(weights)/sqrt(pi);          %  ditto

MaxLoss =80;
Term =5;
delta =0.25;
ValSteps =Term/delta;
RecAssets=Assets(:,1)*(1-Rec);
GreatCommonDivisor= gcd2(RecAssets)
RecAssets=RecAssets/GreatCommonDivisor;

AssetTemp =repmat(RecAssets,1,max(RecAssets));
AssetRange = repmat(min(RecAssets):max(RecAssets),size(RecAssets),1);
result = (AssetTemp==AssetRange)';

M=256;  % Fourier resolution
unconditionalLoss=zeros(MaxLoss+1,ValSteps);

for t =delta:delta:Term
    conditionalLoss=zeros(MaxLoss+1,GaussPoints);

  for k= 1:GaussPoints
     V =  abscissas(k);
      g = zeros(MaxLoss+1,1);
    % Calculate the probabilities and store the vector.
    prob = zeros(1,50);
    prob=  p_Vector(t,rho,V,h)';

     lambda= sum(prob);

    tt1=[0:80]';
    tt3=zeros(1,81)';
    tt2=RecAssets+1;  % because of zero based indexing   sum(tt4(find(tt2==k)),2)
    tt3(2)=sum(prob(find(tt2==2)),2)/lambda;
    tt3(3)=sum(prob(find(tt2==3)),2)/lambda;
    tt3(4)= sum(prob(find(tt2==4)),2)/lambda;
    y=[tt1,tt3];
    x=y(:,2)';  % take probs, and make row vector

   zeropads= length(x);
   f1=[x,zeros(1,M-zeropads)];
   fhat1=fft(f1);
   P1=exp(lambda*(fhat1-1));
  g1=ifft(P1);
  conditionalLoss(:,k) =abs(g1(1:81)');

   % For each time, Gauss-Hermite point, collect the vector

   end % loop for Gaussian Quadrature

   adjustedWeights =  repmat(weights',MaxLoss+1,1);
   unconditionalLoss(:, t/delta) = sum(conditionalLoss.*adjustedWeights,2);

end %  end of time loop
%------------------------------------------------------------------------
Lower =750;
Upper =1500;
lossVector =0:1:MaxLoss;
time = delta:delta:Term;
ExpectedLossTime = sum(unconditionalLoss.*repmat(TrancheLoss(Lower,Upper,
GreatCommonDivisor*lossVector'),1,ValSteps),1);
ExpectedLossTime_lessOne = ExpectedLossTime;
ExpectedLossTime_lessOne(end) =0; %make last  element zero
ExpectedLossTime_lessOne=ExpectedLossTime_lessOne([ end 1:end-1 ]); % shift right/down
1 element

DiffExpectedLossTime =diff([0,ExpectedLossTime]);

timeDefaultLeg = (delta/2):delta:Term;   % 0.5*(T(i+1)+T(i)) ASB page 69

DefaultLeg = sum(exp(-RiskFree*timeDefaultLeg).*DiffExpectedLossTime);
PremiumLeg =delta*sum(exp(-RiskFree*time).*(Upper-Lower -
0.5*(ExpectedLossTime+ExpectedLossTime_lessOne)));
100*DefaultLeg/PremiumLeg
toc
```

REFERENCES

Abramowitz, M, & Stegun, I (1972) *Handbook of Mathematical Functions,* Dover Publications, Toronto

Andersen, L., Sidenius, J & Basu, S. (2003) All your hedges in one basket, *Risk* **11**, 67–72

Andersen, L. & Sidenius, J (unpublished) The Bank of America guide to advanced correlation products, 2004

Arvanitis, A. Browne, C., Gregory, J. & Martin, R. (unpublished) *Credit Loss Distributions and Economic Capital*, Quantitative Credit & Risk Research, Paribas, 1998

Arvanitis, A. & Gregory, J. (2001) *Credit: The Complete Guide to Pricing, Hedging and Risk Management***,** Risk Books, London

Avellaneda, A. & Zhu, J. (2001) Distance to default, *Risk* **12**, 125–129

Basel Committee on Bank Supervision (unpublished) International Convergence of Capital Measurement and Capital Standards: A Revised Framework. Bank for International Settlements, Basel, 2004

Baxter, M. & Rennie, A. (1996) Financial Calculus, Cambridge University Press, Cambridge

Belov, A.G. & Galkin, V.Ya. (1991) Algorithms for the numerical evaluation of the compound Poisson distribution, *Computational Mathematics and Modeling* **2,** 3, 261–267

Bielecki, T.R. & Rutkowski, M. (2002) *Credit Risk: Modeling, Valuation and Hedging*, Springer-Verlag, Berlin

Black, F. & Cox, J.C. (1976) Valuing corporate securities: Some effects of bond indenture provisions. *Journal of Finance* **31**, 351–367

Black, F. & Scholes, M. (1973) The pricing of options and corporate liabilities, *Journal of Political Economy* **81**, 637–654

Bluhm, C., Overbeck, L. & Wagner, C. (2003) *An Introduction to Credit Risk Modeling,* CRC Press, Boca Raton

Bluhm, C. & Wagner, C. (2004) Semi-analytic approaches to CDO modelling*, Economic Notes* **33**, 2, 233–255

Boscher, H. & Ward, I. (2002) Long or short in CDOs, *Risk* **6**, 125–129

Boyle, P., Broadie, M. & Glasserman, P. (1997) Monte Carlo methods for security pricing, *Journal of Economic Dynamic and Control* **21**, 1267–1321

Bracewell R.N. (2000) *The Fourier Transform and Its Applications*, 3rd Ed, Singapore, McGraw-Hill

Brandimarte, P. (2006) Numerical *methods in finance and economics: a MATLAB-based introduction,* Wiley, New Jersey

British Bankers' Association (unpublished) Credit Derivatives Report, 2006

Cerny, A. (2004) *Mathematical Techniques in Finance*, Princeton University Press, New Jersey

Cheng, L. (unpublished) Analysis and Numerical Solution of an Inverse First Passage Problem from Risk Management, Thesis, University of Pittsburgh, 2005

Cheng, L., Chen, X., Chadam, J. & Saunders, D. (unpublished) Numerical computation of first-crossing boundary problem, 2005, Working Paper

Cherubini, U., Luciano, E & Vecchiato, W. (2004) Copula Methods in Finance, Wiley, West Sussex

Chung, K.L. (2001) *A Course in Probability Theory*, Academic Press, San Diego

Gupton G. M., Finger C., Bhatia M. (unpublished) CreditMetrics: The Benchmark for Understanding Credit Risk, JP Morgan, Inc.,1997

DeBuysscher, A. & Szegö, M (unpublished) The Fourier transform method – Technical document, Moodys, 2003

De Prisco, B., Iscoe, I. & Krenin, A. (2005) Loss in Translation, *Risk* **6**, 77–82

Duffie, D. & Singleton, K. (1999) Modeling the term structures of defaultable bonds, *Review of Financial Studies* **12**, 4, 197–226

Duffie, D. & Singleton, K. (2003) *Credit Risk Pricing Measurement and Management*, Princeton series in Finance, New Jersey

Durret, R (1996) *Probability Theory and Examples,* Wadsworth Publishing Company, Belmont

Dutang, C., Goulet, V. & Pigeon, M. (2008) actuar : An R package for actuarial science, *Journal of Statistical Software* **25**,7

Embrechts, P. & Frei, M. (2007) Panjer recursion versus FFT for compound distributions, Published online and to appear in *Mathematical Methods in Operations Research*

Feller, W. (1971) *An Introduction to Probability Theory and Its Applications*, Volume I, Wiley, New York

Finger, C. (unpublished) A. Comparison of stochastic default rate models, RiskMetrics*,* 2000

Frees, J., Carriere, J. & Valdez, E. (1996) Annuity valuation with dependent mortality. *Journal of Risk and Insurance* **63,** 229–61

Fusai, G. & Roncoroni, A. (2008) *Implementing Models in Quantitative Finance*, Springer, Heidelberg

Gibson, M.S. (unpublished) Understanding the risk of synthetic CDOs, Federal Reserve Board, Working paper, 2004

http://www.federalreserve.gov/pubs/feds/2004/200436/200436pap.pdf

Glasserman, P. (2004) *Monte Carlo Methods in Financial Engineering*, Springer-Verlag, New York

Glasserman, P. & Ruiz-Mata, J. (2006) Computing the credit loss distribution in the Gaussian copula model: a comparison of methods, *Journal of Credit Risk* **2** 4, 33–66.

Goodman, L.S. & Fabozzi F.J. (2002) *Collateralized Debt Obligations*, Wiley Finance, New York

Gordy, M. B. (2003) A risk-factor model foundation for ratings-based bank capital rules. *Journal of Financial Intermediation* **12**, 199 - 232.

Greenberg, A., O'Kane, D. & Schloegl, L. (unpublished) LH+: A fast analytical model for CDO hedging and risk management, Quantitative Credit Research Strategy, Lehman Brothers, 2004

Gupton, G.M., Finger, C.C. & Bhatia, M. (unpublished) Creditmetrics - Technical document, JP Morgan & Co. Incorporated, 1997

http://www.riskmetrics.com/pdf/CMTD1.pdf

Hamilton, D.T., Varma, P.,Ou, S. and Cantor, R. (unpublished) Default & Recovery Rates of Corporate Bond Issuers 1920-2005, Moody's Investors Service, January, 2006

Hattendorf, K. (1868) Ueber die berechnung der rcscrven und des risico bei der lebensversicherung. In Rundschauder Versicherungen Volume XVIII, Leipzig

Heath, D., Jarrow, R.A. & Morton, A. (1992) Bond pricing and the term structure of interest rates: A new methodology, *Econometrica* **60**, 77–105

Higham, N.J. (1996) Accuracy and Stability of Numerical Algorithms, Siam, Philadelphia

Hull, J.C. & White, A. (2001) Valuing credit default swaps II: Modelling default correlations, *Journal of Derivatives* **2**, 1, 7–16

Hull, J.C. (2003) *Options, Futures, and Other Derivatives*, Prentice Hall, New Jersey

Hull, J.C. & White, A. (2004) Valuation of a CDO and nth to default swap without Monte Carlo simulation, *Journal of Derivatives* **12**, 8–23

Iscoe, I., Krenin, A. & Rosen, D (unpublished) An Integrated Market and Credit Risk Portfolio. Model, Algorithmics Research Quarterly, 21–38, 1999

Iscoe, I., & Krenin, A. (unpublished a) Default boundary problem, Algorithmics, working paper, 2002

Iscoe, I., & Krenin, A. (unpublished b) Pricing and valuation of synthetic CDOs, Algorithmics, working paper, 2004

Jarrow, R. & Turnbull, S. (1992) Credit risk: Drawing the analogy *Risk* **5,** 9, 63–70

Jarrow, R. & Turnbull, S. (1995) Pricing of derivatives on financial securities subject to credit risk, *Journal of Finance* **50,** 1, 53–85

Kalemanova, A., Schemid B, & Werner, R. (unpublished) The Normal Inverse Gaussian distribution for synthetic CDO pricing, Risklab, 2005

http://www.mathfinance.ma.tum.de/personen/KalemanovaSchmidWerner.pdf

Kelley C.T. (1995) *Iterative Methods for Linear and Nonlinear Equations*, Society for the Industrial and Applied Mathematics, Philadelphia

Klugman, S.A., Panjer, H.H., & Willmot, G.E. (2004) *Loss Models: from data to decisions,* Wiley Interscience, Holboken

Kohl, M., Ruckdeschel, P. & Stabla, T. (unpublished) General Purpose Convolution Algorithm for Distributions in S4-Classes by means of FFT, Universität Bayreuth, 2005

Laurent, J.P. & Gregory, J. (2003) I will survive *Risk* **6**, 103–107

Li, D.X. (1998) Constructing a credit curve, *Risk Credit Risk Special Report* **11**, S40–S44.

Li, D.X. (2000) On default correlation: A copula function approach, *Journal of Fixed Income* **9***,* March, 43–54

Li, D.X. (2006) Editorial: It's all about credit, North American Actuarial Journal **10,** 1, iii–iv

Loève, M., (1977) *Probability Theory I*, 4th ed., Springer-Verlag, Michigan

Lukacs, E. (1970) *Characteristic Functions*, 2nd ed., Griffin, London

Luo, L.S.J. (2005) Bootstrapping default probability curves, *Journal of Credit Risk* **1**, 4, 169–179

Lütkebohmert, E. (2009) *Concentration Risk in Credit Portfolios*, Springer-Verlag, Heidelberg

Marshall, R., Naldi, M. & Tejwani, G. (unpublished), The implications of implied correlation, Quantitative Fixed Income research, Lehman Brothers, 2004

McLeod, A., Panayiotis, T. & Gupta, S. (2005) Time for multi-period capital models, *Risk* **6**, 74–78

Merton, R.C. (1974) On the pricing of corporate debt: The risk structure of interest rates, *Journal of Finance* **29**, 449–470

Mittnik, S., Doganoglu, T. & Chenyao, D. (1999) Computing the probability density function of the stable Paretian distribution, *Mathematical and Computer Modelling* **29**, 235–240

Mood, A., Graybill, F.A. & Boes, D.A. (1974) *Introduction to the Theory of Statistics*, McGraw-Hill, New York

Morokoff, W.J. (2003) Simulation methods for risk analysis of collateralized debt obligations, *Proceedings of the Winter Simulation Conference,* 335–342

Nelson, R.B. (1999) *An Introduction to Copulas,* Springer, New York

O'Kane, D. & Turnbull, S. (unpublished) Valuation of credit default swaps, Lehman Brothers, 2003

Panjer, H.H. (1981) Recursive evaluation of a family of compound distributions, *ASTIN Bulletin* **12**,1, 22–26

Panjer, H.& Willmot, G. (1992) *Insurance Risk Models*, Society of Actuaries, Shaumburg.

Panjer, H.H. (2006) *Operational Risk Modelling Analytics*, Wiley, Hoboken

Paollella, M. (2007) *Intermediate Probability,* Wiley, West Sussex

Petrov, V.V. (1975) *Sums of Independent Random Variables,* Springer, Berlin

Press, W.H., Teukolsky, S.A., Vetterling, W.T. & Flannery, B.P. (1997) *Numerical Recipes: The Art of Scientific Computing,* Cambridge University Press, Cambridge

Quarteroni, A., Sacco, R. & Saleri, F. (2007) *Numerical Mathematics,* Springer, Heidelberg

Robertson, J. (1992) The computation of aggregate loss distributions, *Proceedings of the Casualty Actuarial Society* **79**, 57–133

Rockmore, D.N. (2000) The FFT: An algorithm the whole family can use, *Computing in Science & Eng* **2**, 1, 60–64

Ross, S.M. (2006) *Simulation,* 4th ed. London, Elsevier Academic Press

Ruckdeschel, P., Kohl, M., Stabla, T. & Camphausen, F. (2006), S4 classes for distributions, *RNews* **6**, 2, 2–6

Samuelson, P. (1965) Rational theory of option pricing, *Industrial Management Review,* **6**, 2, 13–39

Schaller, P. & Temnov, G (2008) Efficient and precise computation of convolutions: applying FFT to heavy tailed distributions, *Computational Methods in Applied Mathematics*, **2**, 187–200

Schönbucher, P.J. (2003) *Credit Derivatives Pricing Models* Wiley Finance, West Sussex, England

Sklar, A. (1959) Fonctions de répartition à *n* dimensions e leurs marges. *Publications de l'Institut de Statistique de l'Univiversite de Paris* **8,** 229–23

Smithson, C. (2004) Hedging a portfolio of structured credit assets, *Risk* **10**, 66–69

Stuart, A. & Ord, K. (1994) *Kendall's Advanced Theory of Statistics*, Arnold, London

Thomson, R.J. (2004) Editorial: Is actuarial science really a science?, *South African Actuarial Journal*, 97–103

Tudball, D (unpublished) "The Big Bang Started it All", Wilmot Magazine, July, 2003

Vasicek, O.A. (unpublished a) The probability of loss on loan portfolio, MoodysKMV, 1987

Vasicek, O.A. (unpublished b) Limiting loan loss probability distribution, MoodysKMV, 1991

http://www.moodyskmv.com/research/portfoliotheory.html

Vasicek, O.A. (2002) Loan portfolio value, *Risk* **12,** 160–162

Wang, S. (1998) Aggregation of Correlated Risk Portfolios: Models & Algorithms, *Proceedings of the Casualty Actuarial Society* ***LXXXV***, 848–939

Whelan, S.F., Bowie, D.C. & Hibbert, A.J. (2002) A Primer in Financial Economics. *British Actuarial Journal* **8**, 27–74

Wilde,T. (unpublished) CreditRisk: A credit risk management framework, Credit Suisse First Boston, 1997

Yang, W.Y, Cao, W, Chung, T.S. & Morris, J. (2005) *Applied Numerical Methods using Matlab*, John Wiley & Sons, Inc , New Jersey

Zucca, C., Sacerdote, L & Peskir, G. (unpublished) On the inverse first-passage problem for a Wiener process, Universita degli Studi Torino, 2003

http://www.dm.unito.it/quadernidipartimento/2003/pdf/q02-03.pdf