ALGORITHMS FOR IMPROVING THE EFFICIENCY OF CEV, CIR AND JDCEV OPTION PRICING MODELS

Pedro Filipe Botelho Negrão de Sousa

Dissertation submitted as partial requirement for the conferral of

Master in Finance

Supervisor:
Doutor José Carlos Dias, Professor Auxiliar com Agregação, ISCTE-IUL Business School, Departamento de Finanças

May 2017

Abstract

The non-central chi-square distribution function has extensive use in the field of Mathematical Finance. To a great extent, this is due to its involvement in the *constant elasticity of variance* (hereafter, CEV) option pricing model of Cox (1975), in the term structure of interest rates model of Cox *et al.* (1985a) (hereafter, CIR), and the *jump to default extended CEV* (hereafter, JDCEV) framework of Carr and Linetsky (2006). Efficient computation methods are required to rapidly price complex contracts and calibrate financial models. The processes with several parameters, like the CEV or JDCEV models that we will address are examples of where this is important, since in this case the pricing problem (for many strikes) is used inside an optimization method. With this work we intend to test recent developments concerning the efficient computation of the non-central chi-square distribution function in the context of these option pricing models. We will give particular emphasis to the recent developments presented in the work of Gil *et al.* (2012), Gil *et al.* (2013), Dias and Nunes (2014), and Gil *et al.* (2015). For each option pricing model, we will define reference data-sets compatible with the most common combination of values used in pricing practice, following a framework that is similar to the one presented in Larguinho *et al.* (2013). We will conclude by offering novel analytical solutions for the JDCEV delta hedge ratios for the recovery parts of the put.

Keywords: Option pricing, JDCEV model, Special functions, Algorithms.
JEL Classification: G12, C63.

Sumário

A distribuição de probabilidade chi-quadrado não-central tem sido alvo de vasta utilização no domínio da Matemática Financeira, em grande parte devido à sua utilização no modelo *constant elasticity of variance* (doravante, CEV) de Cox (1975), no *term structure of interest rates model* de Cox *et al.* (1985a) e no modelo *jump to default extended CEV* (doravante, JDCEV) de Carr and Linetsky (2006). Métodos de cálculo eficientes e rápidos são de especial relevância na calibração de modelos para a determinação do preço de contratos financeiros complexos. Os modelos CEV, CIR e JDCEV são exemplos de modelos com diversos parâmetros que, quando usados em contexto de determinação do preço de opções com vários preços de exercício, mostram como esta optimização é fundamental. Com este trabalho pretendemos testar os mais recentes desenvolvimentos no cálculo eficiente da distribuição de probabilidade não-central chi-quadrado, no contexto dos modelos de cálculo de preço de opções mencionados anteriormente. Daremos ênfase aos recentes desenvolvimentos apresentados nos trabalhos de Gil *et al.* (2012), Gil *et al.* (2013), Dias and Nunes (2014) e de Gil *et al.* (2015). Para cada um dos modelos, definiremos um conjunto de parâmetros de referência compativel com as combinações mais usadas na prática, seguindo uma metodologia similiar à usada em Larguinho *et al.* (2013). Concluímos com a derivação de novas soluções analíticas para os rácios de *delta hedging* no modelo JDCEV.

Palavras-chave: Preço de opções financeiras, Modelo JDCEV, Funções especiais, Algoritmos.
Sistema de Classificação JEL: G12, C63.

Acknowledgements

I express my gratitude to Professor José Carlos Dias for his precious insightfulness and encouragement throughout this work. It was a privilege to work under his guidance.

I thank my wife Sara, for her unconditional love and compassion for the world.

I thank my son Afonso, for his life-filling and joyful expression.

I thank my parents for their never-ending support.

"L'espérance mathématique du spéculateur est nulle"

Louis Bachelier

Contents

Chapter 1

Introduction

The non-central chi-square distribution function has extensive use in the field of Mathematical Finance. To great extent, this is due to its involvement in the *constant elasticity of variance* (hereafter, CEV) option pricing model of Cox (1975), in the term structure of interest rates model of Cox *et al.* (1985a) (hereafter, CIR), and the *jump to default extended CEV* (hereafter, JDCEV) framework of Carr and Linetsky (2006). With this work, we test the recent developments concerning the efficient computation of the non-central chi-square distribution function in the context of the option pricing models referred before. We will give particular emphasis to the recent work of Gil *et al.* (2012), Gil *et al.* (2013), Dias and Nunes (2014), Gil *et al.* (2014) and Gil *et al.* (2015) in parallel with Sun *et al.* (2010) and Kapinas *et al.* (2009). We expect to conclude about the efficiency and accuracy of these algorithms when compared to the previously available methods in the context of option pricing models we are addressing.

When pricing financial options under the aforementioned models, the incomplete Gamma function is important. It is used in the computation of the non-central chi-square distribution function used in the CEV and CIR models and also in the truncated and raw moments for option pricing under the JDCEV model. For this purpose, we will test the recent developments introduced by Gil *et al.* (2012) work.

The non-central chi-square or the non-central Gamma cumulative distribution function, also known as Marcum-Q function, plays a central role in the computation of option prices under the CEV and CIR option pricing models. For this purpose, we will test the recent developments introduced by Gil *et al.* (2015) against Benton and Krishnamoorthy (2003).

The truncated and raw moments of the non-central chi-square distribution function, also known as Nuttall-Q functions, play a key role in the computation of option prices under the JDCEV pricing model. For this purpose, we will test the recent developments introduced by Dias and Nunes (2014) against Sun *et al.* (2010) and Gil *et al.* (2013).

Efficient computation methods are required to rapidly price complex contracts and calibrate financial models. For the calibration process, i.e., when fitting model parameters of the stochastic asset processes to market data, we normally need to price European options at a single spot price, very quickly, with varying strike prices. Examples of where this is important are the processes with several parameters, like the CEV or JDCEV model that we will address, since there the pricing problem (for many strikes) is used inside an optimization method. As stated in Broadie and Detemple (1996), a trader wishing to price a single option requires a computation speed on the order of 1 second. However, dealers or large trading desks may need to price thousands of options on an hourly basis, considering that higher accuracy is known to always be better, except if insignificant price improvements are obtained at an unacceptable cost in terms of computation time.

To achieve the proposed goals, we plan to thoroughly describe both the aforementioned option pricing models and the recent developments in the computation of the non-central chi-square distribution function. We plan to implement the option pricing models and the non-central chi-square algorithms in Matlab and Fortran programming languages.

Regarding the overall methodology, we plan to follow a general structure similar to the one presented in Larguinho *et al.* (2013) that, following Broadie and Detemple (1996), randomly generate the option pricing parameters according to probability distributions for the pricing parameters.

The selected algorithms used to compute the non-central chi-square distribution function will be tested for speed-accuracy trade-off in the context of option pricing under CEV, CIR and JDCEV, to obtain a comparison framework as the one presented in Larguinho *et al.* (2013, Table 2).

In general, for each algorithm, we will register the maximum absolute error (MaxAE), the maximum relative error (MaxRE), the root mean absolute error (RMSE), the mean absolute error (MeanAE), the number of times the absolute difference between the two methods (algorithm and benchmark) exceed a pre-determined difference (K), and the computation time in seconds. With this setup we can test for both speed and accuracy.

The remainder of the work proceeds as follows. In Chapter 2 we describe the CEV option pricing model. In Chapter 3 we describe the CIR term structure of interest rates model. In

Chapter 4 we describe the JDCEV option pricing model. In Chapter 5 we characterize the non-central chi-square distribution function, the Marcum and Nuttall functions and their relations with the non-central chi-square function. We review the algorithms involved in their implementation in the context of the models previously enumerated. In Chapter 6 we present the numerical results of the tests we have performed. In Chapter 7 we offer offer new analytical solutions for the JDCEV delta hedge ratios for the recovery parts of the put. Chapter 8 concludes.

Chapter 2

CEV Option Pricing Model

The CEV model of Cox (1975) offers a notable improvement over the seminal work of Black and Scholes (1973) and Merton (1973) (hereafter, BSM) option pricing model, that assumes that the underlying asset price is governed by a geometric Brownian motion with constant volatility. The log-normal diffusion process of the BSM model was oftentimes challenged in favor of a more adequate distributional assumption in accordance with empirical observations. As Jackwerth and Rubinstein (2012) point out, the observed implied risk-neutral probability densities evidence high skewness to the left and are shown to be very leptokurtic, contrasting with the log-normal assumption of the BSM model.

In practice, if we equate the BSM model option price to its market price, we can compute what is commonly known as the option *implied volatility*. In empirical data, we can observe that this volatility is not constant and it varies with the strike price originating an effect that is known as *implied volatility skew* — see, for example, Dennis and Mayhew (2002). Another significant observation first discussed by Black (1976) is the so-called *leverage effect*. It stems from the empirical evidence that stock price level is negatively correlated with the realized stock volatility — see, for example, Bekaert and Wu (2000).

The CEV model offers the flexibility to be consistent with empirical observations and overcome the BSM model drawbacks described earlier and, at the same time, it offers a closed-form solution to price financial options. Even though the complete derivation of the Cox (1975) option pricing formulas is outside the scope of this work — for details, see, for instance, Hsu *et al.* (2008) — we will consider some aspects of the process that are key for a better understanding of the mechanics of CEV.

2.1 CEV diffusion process

In the CEV diffusion process of Cox (1975), assuming the equivalent martingale measure $\mathbb{Q}$ (risk-neutral probability measure) as given, the asset price $\{S_t, t \geqslant 0\}$ is governed by the following stochastic differential equation,

$$\mathrm{d}S_t = S_t\,\mathrm{d}t + S_t^{\beta/2}\,\mathrm{d}W_t^{\mathbb{Q}}, \quad t \geq 0, \quad S_0 = S > 0, \tag{2.1}$$

with *local volatility function* defined by,

$$\sigma(S_t) = \delta S_t^{(\beta/2)-1}, \tag{2.2}$$

for δ, $\beta \in \mathbb{R}$, and where $W_t^{\mathbb{Q}}$ is a standard Wiener process defined on a filtered probability space $(\boldsymbol{\Omega}, \mathcal{F}, \{\mathcal{F}_t\}_{t\geqslant 0}, \mathbb{Q})$ and μ is a constant, representing the risk-neutral drift rate ($\mu = r - q$, being $r \geq 0$ the constant risk-free interest rate and $q \geq 0$ the constant dividend yield).

The elasticity of the local volatility function is $\beta - 2$ given that the variance with respect to price has the relationship $\mathrm{d}\upsilon(S_t)/\upsilon(S_t) = (\beta - 2)\mathrm{d}S_t/S_t$ that, upon integration on both sides, yields the instantaneous variance of asset returns $\upsilon(S_t) = \sigma^2 S^{\beta-2}$. This implies that the elasticity of the variance is independent of the asset price, given the proportionality of the volatility to the power of the underlying asset price.

As particular cases, the CEV specification of (2.1) encompasses the log-normal geometric Brownian motion of Black and Scholes (1973) and Merton (1973) ($\beta = 2$) and the absolute diffusion ($\beta = 0$) and square-root processes ($\beta = 1$) of Cox and Ross (1976). For the case of $\beta < 2$ ($\beta > 2$) the *local volatility function* of (2.2) becomes a decreasing (increasing) function of the asset price. The δ parameter, assumed to be positive, is defined as the *scale parameter* of the *local volatility function*, defining the initial instantaneous volatility at time $t = 0$, $\sigma_0 = \sigma(S_0) = \delta S_0^{\beta/2-1}$.

The case of $\beta < 2$ was originally studied by Cox (1975) and later extended to $\beta > 2$ by Emanuel and MacBeth (1982). Although Cox initially restricted the elasticity parameter to $0 \leqslant \beta \leqslant 2$, evidence has been found that β is generally smaller than 2 — see MacBeth and Merville (1980) — and that typical values of β implicit in the S&P500 in the post-crash of 1987 could be as low as $\beta = -6$ — see Jackwerth and Rubinstein (2012). Jackwerth and Rubinstein call the model with $\beta < 0$ the *unrestricted CEV*. Empirical evidence can be found in the literature for the case of $\beta < 2$ (with downward sloping implied volatility or *direct leverage effect*) to be of relevance for the stock index option market — see, for instance, Black (1975) and MacBeth and Merville (1979) — and that values of $\beta > 2$ (with downward sloping implied volatility or *inverse leverage effect*) could be expected for some commodity futures options — see, for instance, Davydov and Linetsky (2001), Geman and Shih (2009) and Dias and Nunes (2011).

2.2 CEV diffusion transition probability function

The option valuation problem is intrinsically related with the probability distribution of the terminal stock value. According to Cox and Ross (1976), the issue of option pricing is, in fact, equivalent to determining the distribution of the stock variable S and, hence, the distribution underlying the stochastic differential equation assumed to govern the movement of the asset. Cox and Ross used a hedging argument to propose a framework where risk-neutrality is the choice of preferences and where, if so, the expected return on the stock is the same as in the options'. For the stock,

$$\mathbb{E}\left\{\frac{S_T}{S_t}\middle|S_t\right\} = e^{r(T-t)}, \tag{2.3}$$

and for the the general European option with boundary value, $P(S,T) = h(S)$, then, at time t,

$$\mathbb{E}\left\{\frac{P(S_T,T)}{P(S,t)}\middle|S_t\right\} = \frac{1}{P(S,t)}\,\mathbb{E}\left\{h(S_T)|S_t)\right\} = e^{r(T-t)}, \tag{2.4}$$

or

$$\begin{aligned} P(S,t) &= e^{-r(T-t)}\,\mathbb{E}\left\{h(S_T)|S_t\right\} \\ &= e^{-r(T-t)}\int h(S_T)\mathrm{d}F(S_T,T|S_t,t), \end{aligned} \tag{2.5}$$

where $F(S_T,T|S_t,t)$ represents the probability distribution of the stock at time T, S_T, given the stock price at time $t < T$, S_t. From (2.5), it becomes clear that if we know the cumulative probability distribution of the stock, we can value the option.

In the CEV model, for $\beta = 2$ we are in presence of the log normal-diffusion of Black-Scholes and so, the transition probability density function comes down to a normal density function with mean m and variance V. If $\beta \neq 2$ the transition is much more complex. First Cox (1975) for $\beta < 2$ and later Emanuel and MacBeth (1982) for $\beta > 2$, derived the following transition probability function [1]:

$$f(S_T|S_t, T > t) = \begin{cases} (2-\beta)k^{1/(2-\beta)}(xy^{1-2\beta})^{1/(4-2\beta)} \\ \times e^{-x-y} I_{1/2-\beta}(2(xy)^{1/2}) \Leftarrow \beta < 2 \\ (\beta-2)k^{1/(2-\beta)}(xy^{1-2\beta})^{1/(4-2\beta)} \\ \times e^{-x-y} I_{1/\beta-2}(2(xy)^{1/2}) \Leftarrow \beta > 2 \end{cases}, \tag{2.6}$$

where,

$$k = \frac{2(r-q)}{\delta^2(2-\beta)\left[\mathrm{e}^{(r-q)(2-\beta)\tau} - 1\right]}, \tag{2.7a}$$

$$x = kS_t^{2-\beta} e^{(r-q)(2-\beta)\tau}, \tag{2.7b}$$

$$y = kS_T^{2-\beta}, \tag{2.7c}$$

$$\delta^2 = \sigma_0^2 S_0^{2-\beta}, \tag{2.7d}$$

$$\tau = T - t, \tag{2.7e}$$

and where r denotes the risk-free interest rate, q denotes the continuous proportional dividend rate and $I_q(.)$ is the *modified Bessel function* of the first kind of order q, given, for instance, in Abramowitz and Stegun (1972, Eq. 9.6.10).

2.3 CEV pricing solutions to European-style options

Given the transition probability functions shown in (2.6), the European option formula can be derived by taking the conditional expectation on the risk-neutralized process of the stock price according to the Cox-Ross pricing equation of (2.5).

First Cox for $\beta < 2$, and later Emanuel and Macbeth for $\beta > 2$, derived the following option pricing formulas, in terms of the *standard complementary Gamma distribution*

[1] For more details on the derivation of the pricing solutions, see, for instance, Chen and C.-F. Lee (2010) and the references therein to Feller (1951) and Breiman (1986) concerning the standard procedure to identify the transition density, if it exists.

function:

$$c_t(S_t, X, T) = \begin{cases} S_t e^{-r\tau} \sum_{n=0}^{\infty} \frac{e^{-x} x^n G(n+1+1/(2-\beta), kX^{2-\beta})}{\Gamma(n+1)} \\ -X e^{-r\tau} \sum_{n=0}^{\infty} \frac{e^{-x} x^{n+1/(2-\beta)} G(n+1, kX^{2-\beta})}{\Gamma(n+1+1/(2-\beta))} \Leftarrow \beta < 2 \\ S_t e^{-r\tau} \left[1 - \sum_{n=0}^{\infty} \frac{e^{-x} x^{n+1/(\beta-2)} G(kX^{2-\beta}, n+1)}{\Gamma(n+1+1/(\beta-2)))}\right] \\ -X e^{-r\tau} \left[1 - \sum_{n=0}^{\infty} \frac{e^{-x} x^n G(kX^{2-\beta}, n+1+1/(\beta-2))}{\Gamma(n+1)}\right] \Leftarrow \beta > 2 \end{cases}, \tag{2.8}$$

where $G(m, v) = [\Gamma(m)]^{-1} \int_v^{\infty} e^{-u} u^{m-1} \mathrm{d}u$ is the *standard complementary Gamma distribution function* and k, x and τ are as defined in (2.7a), (2.7b) and (2.7e) respectively.

Schroder (1989) expressed the CEV model in terms of the non-central chi-square distribution as follows:

$$c_t(S_t, X, T) = \begin{cases} S_t \mathrm{e}^{-q\tau} \, Q(2y; 2 + \frac{2}{2-\beta}, 2x) - X \mathrm{e}^{-r\tau} \\ \times \left[1 - \, Q(2x; \frac{2}{2-\beta}, 2y)\right] \Leftarrow \beta < 2 \\ S_t \mathrm{e}^{-q\tau} \, Q(2x; \frac{2}{\beta-2}, 2y) - X \mathrm{e}^{-r\tau} \\ \times \left[1 - \, Q(2y; 2 + \frac{2}{\beta-2}, 2x)\right] \Leftarrow \beta > 2 \end{cases}, \tag{2.9}$$

where $Q(\omega, v, \lambda)$ is the non-central chi-square distribution function evaluated at ω, with v degrees of freedom and non-centrality parameter λ, and where k, x, y, δ and τ are as defined in (2.7a) to (2.7e).

Although our analysis will be primarily focused on call options, the CEV put option formulae can be expeditely derived with the help of the put-call parity relationship, being the time-t value of an European-style put given by

$$p_t(S_t, X, T) = \begin{cases} X \mathrm{e}^{-r\tau} \, Q(2x; \frac{2}{2-\beta}, 2y) - S_t \mathrm{e}^{-q\tau} \\ \times \left[1 - Q(2y; 2 + \frac{2}{2-\beta}, 2x)\right] \Leftarrow \beta < 2 \\ X \mathrm{e}^{-r\tau} \, Q(2y; 2 + \frac{2}{\beta-2}, 2x) - S_t \mathrm{e}^{-q\tau} \\ \times \left[1 - Q(2x; \frac{2}{\beta-2}, 2y)\right] \Leftarrow \beta > 2 \end{cases}, \tag{2.10}$$

and where k, x, y, δ and τ are as defined in (2.7a) to (2.7e).

Chapter 3

CIR Option Pricing Model

The *term structure of interest rates* has long been a matter of great interest for economists. The relationship among the yield of default-free securities and their term to maturity represents a central topic in financial research. The need to price and hedge interest rate contingent claims has played a major role in the need to better understand and to model the behavior of the term structure of interest rates.

Albeit early research in the area is vast[1], there seems to be consensus that it can be identified as belonging to one of two different strands of though, namely, the *Expectations Theory* and the *Market Segmentation Theory* — see, for instance, Fabozzi and Mann (2005).

The *Expectations Theory*, in its broadest interpretation called the *Pure Expectations Theory* — rooting back to, at least, I. Fisher (1896) — states that implied forward rates represent expected future rates — see Lutz (1940). However, this theory does not account for the *price risk* involved, for instance, in investing in a strategy comprising bonds with maturity longer than the holding period. To account for that, Hicks (1939)[2] introduced the so-called *Liquidity Preference Theory*, that builds on the idea that investors would hold longer-term maturities if offered a risk premium, uniform and increasing with maturity, over the expected average future rates. Yet another theory, proposed by Modigliani and Sutch (1966), is known to be the *Preferred Habitat Theory* that, building upon the previous interpretations, rejects

[1]See, for instance, I. Fisher (1896), Macaulay (1938), Hicks (1939), Lutz (1940), Modigliani and Sutch (1966), Malkiel (1966), Telser (1967), Nelson (1972), Modigliani and Shiller (1973), J. W. Elliott and Baier (1979), Shiller (1979) and Shiller (1981), Cox *et al.* (1981), Brennan and Schwartz (1982), Fama (1984a) and Fama (1984b), Cox *et al.* (1985b) and Cox *et al.* (1985a) and Shiller and McCulloch (1987).

[2]Building on the work of Keynes (1930) and Keynes (1936).

Hicks's ever-rising *price risk* with maturity. Instead, it asserts that *price risk* can be positive or negative, to accommodate imbalances in the demand and supply of funds across different terms, forcing investors to shift maturities, thus having to compensate for either price or reinvestment risk.

The *Market Segmentation Theory* suggested by Culbertson (1957), states that the shape of the term structure is constrained by the asset/liability management by borrowers and creditors, in specific maturity sectors. This theory, contrasting with the *Expectations Theory*, does not consider the possibility of market participants shifting maturities, to take advantage of differences between expectations and forward rates.

The seminal work of O. Vasicek (1977)[3], based on an *economic equilibrium* approach, introduced the stochastic modeling of the evolution of the *term structure of interest rates* in continuous time. In his model, the continuously compounded interest rates evolve as an Ornstein-Uhlenbeck process with constant coefficients, leading to a positive probability for negative rates. Noteworthy, its analytical tractability, characterized by a Gaussian density short rate process, is hardly surpassed by other distribution models.

Cox *et al.* (1985a) (hereafter, CIR) model implies continuously compounded positive interest rates, characterized by a non-central chi-square distribution. In the CIR model, the volatility is proportional to the square root of the short rate, meaning that if the rate approaches zero, the volatility becomes very small, letting the drift dominate the process, pushing it towards the mean. This is a remarkable improvement over Vasicek's model. Both models belong to the category of what has came to be known as *endogenous* models, given the fact that the term structure is an output rather than an input of the model. This can be seen as a drawback of these models since they cannot be fitted to a currently observed term structure in the market.[4] Hull and White (1990) adapted Vasicek's model to allow for the fit of the current term structure by calibrating a time-dependent drift term, in what has came to be known as the extended Vasicek model. The authors also propose an extension to the CIR model considering time dependent coefficients.

By assuming different processes for the dynamics of the short rate, other *short rate models* were introduced by authors including Cox (1975), Cox and Ross (1976), Black (1976), Merton (1973), Brennan and Schwartz (1977), Dothan (1978), Cox *et al.* (1980), Brennan

[3]Following Black and Scholes (1973) arguments to derive an arbitrage-free price for interest rate derivatives, accounting for the non-tradable feature of interest-rates, under the real-world measure — see Brigo and Mercurio (2006).

[4]Both models are also know to belong to the class of the so-called *affine term-structure models*, deriving from the fact that the continuously-compounded spot rate is an affine function of the short rate — see Duffie and Kan (1996).

and Schwartz (1980), Rendleman and Bartter (1980), Ball and Torous (1983), F. Longstaff (1989), Courtadon (1982), Black *et al.* (1990), Black and Karasinski (1991), Ingersoll Jr. and Ross (1992), Chan *et al.* (1992), Miltersen *et al.* (1997) and Mercurio and Moraleda (2000).

Some authors have developed more complex models, resorting to multi-dimensional analysis, seeking to model the imperfect correlation among different rates in the term structure curve — see, for instance, Cox *et al.* (1985a), Richard (1978), F. Longstaff and Schwartz (1992), Duffie and Kan (1996), Brennan and Schwartz (1979), Schaefer and Schwartz (1987) and Fong and O. A. Vasicek (1991). Jamshidian (1997) has found that two-component diffusions can explain 85% to 90% of the variations in the zero-coupon curve.

Ho and S.-B. Lee (1986) introduced a discrete-time model describing the whole dynamics of the yield curve. Heath *et al.* (1992) (hereafter, HJM), building on the work of Ho and Lee, developed a complete continuous-time *framework* for the stochastic evolution of the complete term structure, relying in the modeling of the instantaneous forward rates, under an arbitrage-free argument. One of the remarkable features of the HJM framework is that virtually any exogenous term-structure model can be derived under its assumptions.

Another popular and promising family of interest-rate models are the so-called *market models*. The log-normal forward-LIBOR model (LFM) — see Miltersen *et al.* (1997) and Brace *et al.* (1997) — and the log-normal forward-swap model (LSM) — see, Jamshidian (1997) — represent interest rate dynamics compatible with Black (1976) formula for the very active interest-rate-options market of *caps* and *swaptions*.

Jump-diffusion models (JDMs) are used to account for discontinuities in the diffusion processes of the interest rates due, for instance, central banks interventions — see, for example, Merton (1976) and Glasserman and Merener (2001) and Glasserman and Merener (2003).

Finally, we should add that although the CIR process is mainly used to model interest rates, it found different financial applications such as the modelling of the stochastic volatility of stock prices — see Heston (1993) — and the credit spread — see Brigo and Alfonsi (2005).

In the following sections, although the complete derivation of the complete CIR framework is outside the scope of this work, we will consider some aspects of the process that are central for a better understanding of the mechanics of CIR.

3.1 CIR diffusion process

The CIR model is a general equilibrium approach, where interest rates are determined by supply and demand, following a logarithmic utility function. The diffusion process, under the risk-neutral process $\mathbb{Q}$, with respect to the risk-adjusted process for the instantaneous interest rate r_t, is governed by the equation,

$$dr_t = [k\theta - (\lambda + k)r_t]\,\mathrm{d}t + \sigma\sqrt{r_t}dW_t^{\mathbb{Q}}, \tag{3.1}$$

where k represents the reversion rate, θ the asymptotic interest rate, σ is the volatility of the process and λ is the market price of the risk parameter. The condition $2k\theta > \sigma^2$ needs to be enforced so that in the process of (3.1), r_t remains positive.

According to Cox *et al.* (1985a), the interest rate dynamics implied in the process has the following relevant empirical properties: (i) Negative interest rates are excluded. (ii) If the interest rate process reaches zero, it can become positive afterwards. (iii) The variance in absolute terms increases when the interest rate increases. (iv) The interest rate has a steady state distribution.

3.2 CIR diffusion transition probability function

The CIR process has an explicitly known transition density function. According to Cox *et al.* (1985a) and Feller (1951), the probability density of the interest rate at time s, conditional on its value at the current time, t, is given by,

$$f(r_s|r_t, s > t) = ce^{-u-v}\left(\frac{v}{u}\right)^{q/2} I_q(2(uv)^{1/2}), \tag{3.2}$$

with,

$$c = \frac{2k}{\sigma^2(1 - e^{-k(s-t)})}, \tag{3.3a}$$

$$u = cr_t e^{-k(s-t)}, \tag{3.3b}$$

$$v = cr_s, \tag{3.3c}$$

$$q = \frac{2k\theta}{\sigma^2} - 1, \tag{3.3d}$$

where $I_q(.)$ is the *modified Bessel function* of the first kind of order q, given, for instance, in Abramowitz and Stegun (1972, Eq. 9.6.10).

3.3 CIR pricing solutions to zero-coupon and coupon bonds

Although our aim is to deal with the CIR option pricing framework and with its use of the non-central chi-square distribution function, for the sake of completeness, we present here the CIR pricing solutions to zero-coupon and coupon bonds.

3.3.1 Zero-coupon bonds

According to Cox *et al.* (1985a), we can write the fundamental equation for the price of a general interest claim $F(r,t)$, with cash flow rate $C(r,t)$,

$$\frac{1}{2}\sigma^2 r\frac{\partial^2 F(r,t)}{\partial r^2}+\kappa(\theta-r)\frac{\partial F(r,t)}{\partial r}+\frac{\partial F(r,t)}{\partial r}-\lambda r\frac{\partial F(r,t)}{\partial r}-rF(r,t)+C(r,t)=0. \quad (3.4)$$

In the CIR framework, the price of a zero-coupon bond, at valuation date t, maturity date at time s (with $s > t$), $Z(r,t,s)$, satisfying the equation with $C(r,t) = 0$, subject to the boundary condition $Z(r,s,s) = 1$, is given by

$$Z(r,t,s) = \mathbb{E}_t^{\mathbb{Q}}\left[e^{-\int_t^s r(u)\mathrm{d}u}\right] = A(t,s)e^{-B(t,s)r}, \quad (3.5)$$

where constants $A(t,s)$, $B(t,s)$, and $\gamma > 0$, are given by

$$A(t,s) := \left[\frac{2\gamma e^{[(\kappa+\lambda+\gamma)(s-t)]/2}}{(\kappa+\lambda+\gamma)(e^{\gamma(s-t)}-1)+2\gamma}\right]^{2\kappa\theta/\sigma^2}, \quad (3.6a)$$

$$B := \frac{2(e^{\gamma(s-t)}-1)}{(\kappa+\lambda+\gamma)(e^{\gamma(s-t)}-1)+2\gamma}, \quad (3.6b)$$

$$\gamma := \left[(\kappa+\lambda)^2+2\sigma^2\right]^{1/2}. \quad (3.6c)$$

3.3.2 Coupon-paying bonds

A coupon bond can be considered a portfolio of zero-coupon bonds with different maturities. That implies that the value of a riskless coupon bond, at the valuation date t and maturity date s (with $s > t$), $P(r,t,s)$, can be expressed as a weighted sum of zero-coupon bond prices, as

$$p(r, t, s) = \sum_{i=1}^{N} a_i Z(r, t, s_i), \tag{3.7}$$

where s_1, s_2, ..., s_N represent the N dates on which payments are made, and each $a_i > 0$ term denotes the amount of the payments made.

3.4 CIR pricing solutions to European-style bond options

3.4.1 Zero-coupon bond options

The CIR model provides solutions for the price of European call and put options, $c^{zc}(r, t, T, s, K)$, with valuation date t, expiration date T, strike price K, with maturity date s (with $s > T > t$), and with instantaneous interest rate r_t. According to Cox *et al.* (1985a), by taking the relevant expectations, and considering the basic valuation equation with terminal condition

$$C(r, t, T; s, K) = \max\left[P(r, T, s) - K, 0\right], \tag{3.8}$$

where $s \geqslant T \geqslant t$, and K restricted to be less than $A(T, s)$ – the maximum bond price at expiration[5] – we reach the following call option pricing formula

$$c^{zc}(r, t, T, s, K) = Z(r, t, s)F(x_1; a, b_1) - KZ(r, t, T)F(x_2; a, b_2), \tag{3.9}$$

where $F(a; a, b)$ represents the non-central chi-square distribution function with a degrees of freedom and non-centrality parameter b,

$$x_1 := 2r^* \left[\phi + \psi + B(T, s)\right], \tag{3.10a}$$

$$x_2 := 2r^* \left[\phi + \psi\right], \tag{3.10b}$$

$$a := \frac{4k\theta}{\sigma^2}, \tag{3.10c}$$

$$b_1 := \frac{2\phi^2 r e^{\gamma(T-t)}}{\phi + \psi + B(T, s)}, \tag{3.10d}$$

$$b_2 := \frac{2\phi^2 r e^{\gamma(T-t)}}{\phi + \psi}, \tag{3.10e}$$

[5]Otherwise the option would never be exercised and would be worth nothing.

where,

$$\phi := \frac{2\gamma}{\sigma^2(e^{\gamma(T-t)} - 1)}, \tag{3.11a}$$

$$\psi := \frac{k + \lambda + \gamma}{\sigma^2}, \tag{3.11b}$$

$$r^* := \left[\ln\left(\frac{A(T,s)}{K}\right)\right] / B(T,s), \tag{3.11c}$$

and r^* represents the critical interest rate below which exercise will occur, *i.e.*, $K = Z(r^*, T, s)$. The CIR corresponding put option on zero-coupon bonds, $p^{zc}(r, t, T, s, K)$, can be expeditely derived with the help of the put-call parity relationship,

$$p^{zc}(r, t, T, s, K) = KZ(r, t, T)Q(x_2; a, b_2) - Z(r, t, s)Q(x_1; a, b_1), \tag{3.12}$$

where $Q(.; a, b)$ represents the complementary non-central chi-square distribution function with a degrees of freedom and non-centrality parameter b.

3.4.2 Coupon-paying bond options

Following the work of Jamshidian (1989), it can be shown that, in all one-factor term structure models, an option on a portfolio of pure discount bonds decomposes into a portfolio of options on the individual bonds. For a portfolio composed of N zero-coupon bonds with different expiry dates s_i, strike price K, maturity date T, we have for an European call option,

$$c^{cb} = (r, t, T, s, K) = \sum_{i=1}^{N} a_i c^{zc}(r, t, T, s_i, K_i), \tag{3.13}$$

with $T < s_1 < s_2 < ... < s_N$, $a_i > 0$, $K_i = Z(r^{**}, T, s_i)$, and where r^{**} is the solution to $\sum_{i=1}^{N} a_i Z(r^{**}, T, s_i) = K$[6]. The corresponding put option on coupon paying bonds, $p^{cb}(r, t, T, s, K)$, can be expeditely derived with the help of the put-call parity relationship,

$$p^{cb} = (r, t, T, s, K) = \sum_{i=1}^{N} a_q p^{zc}(r; t, T, s_i, K_i). \tag{3.14}$$

[6]Alternatively, we could have used the closed-form solution offered by F. Longstaff (1993, Eq. 7).

Chapter 4

Jump to Default Extended CEV Option Pricing Model

Carr and Linetsky (2006) introduced what they called the JDCEV process, an unified framework for the valuation of corporate liabilities, credit derivatives, and equity derivatives as contingent claims, introducing stock-dependent default intensity into Cox (1975) CEV model. When we have addressed the CEV model in Chapter 2, we have pointed out some issues that affect the BSM option pricing model, namely, the *implied volatility skew* effect and the *leverage* effect. In fact, there is another well known phenomenon where empirical evidence deviates from BSM and CEV assumptions. That is the observed positive relationship between equity volatility and default probability.

Several studies demonstrate the aforementioned relationship. Campbell and Taskler (2003) find evidence that cross-sectional variation in bond yields can be well explained by both credit ratings and idiosyncratic firm-level volatility. Cremers *et al.* (2008) show that individual option prices contain important information for credit spreads and contain information on the likelihood of rating migrations. Vassalou and Xing (2004) found that for individual firms positioned in segments with high default risk, equity returns and default risk are positively correlated and default risk seems to be systematic.

Many other studies have focused on the relationship between equity volatility and *credit default swap* (CDS) spreads. For instance, Consigli (2004) documents the positive relationship between stock price volatility implied in option prices and the spread movements for six stocks over 2002-2003. Cremers *et al.* (2008) find that both stock options individual implied volatilities and implied-volatility are influencing factors for credit spreads. Zhang *et al.* (2009) find that volatility risk predicts up to 50% of the CDS spread movement, while when accounting for jump risk or when adding up credit ratings, macroeconomic conditions and firms' balance sheet information, that this fit rises to 69% and to 77%, respectively.

The credit risk modeling can traditionally be interpreted as following two theoretical approaches: the so-called *intensity* or *reduced form* model and the *structural* model. The *structural* class was pioneered by Black and Scholes (1973), Merton (1974) and later extended by Black and Cox (1976) and F. A. Longstaff and Schwartz (1995)[1]. The Merton-Black-Cox-Longstaff-Schartz approach models the firm value evolution and default occurs when the firm market value drops bellow a defined threshold. The *reduced form* approach was studied by, for instance, Jarrow and Turnbull (1995), Jarrow *et al.* (1997), Madan and Unal (1998) and Duffie and Singleton (1999) and considers that default occurs as a pure random event.

Both models are normally considered to be competing and there is debate on which one is the most appropriate — see, for instance, Jarrow (2003) and the references therein. Jarrow and Protter (2004) compare the two approaches arguing that, from an information based perspective, *reduced form* models are preferred to *structural* models since the market does not observe the firm's asset value continuously in time.

Under the diffusion or *structural* model approach, a sudden drop in the value of the firm is impossible and so firms never default by surprise. In the *reduced form* approach, an explicit relation (structural) between default and the firm value is not considered. The hazard rate of default is modeled as an exogenous process, not specifying the economic underpinnings behind the default mechanism. Nevertheless, the *reduced form* JDCEV model is specified in order to provide consistency with the empirical observations described earlier. For the the stock price, it assumes a process with possible diffusion to zero or a jump to default, whichever comes first. Building on the already described properties of CEV — consistency with the *volatility skew* effect and the *leverage* effect — JDCEV further assumes that the default intensity is an increasing affine function of the instantaneous stock variance.

A number of references about defaultable stock models can be found in the literature. First Merton (1976) and later Jarrow and Turnbull (1995) worked in a very tractable framework, producing downward sloping implied volatility skews, extendable to deterministically time varying default arrival rates and instantaneous volatilities. The work of Carr and Linetsky encompasses all the processes previously addressed. The JDCEV relevance is remarkable as it includes killing (default), time-dependent parameters and retains analytical

[1]Introducing stochastic interest rates to address one of the limitations of the Merton (1974) model — see Jarrow (2003) for a review of these limitations.

tractability due to the Bessel processes properties.

4.1 JDCEV diffusion process

The diffusion process modeling the *pre-default* stock price is characterized by the time-inhomogeneous stochastic differential equation

$$\frac{dS_t}{S_t} = [r_t - q_t + \lambda(t, S)]\, dt + \sigma(t, S)\, dW_t^{\mathbb{Q}}, \tag{4.1}$$

with $S_{t_0} > 0$, and where $r_t \geqslant 0$, $r_t \geqslant 0$, $\sigma(S, t) > 0$ and $\lambda(S, t) \geqslant 0$, all time-dependent parameters, represent respectively, the risk-free interest rate, the dividend yield, the instantaneous stock volatility and the default intensity, where the latter two can also be state dependent. The authors consider the probability space $(\Omega, \mathcal{G}, \mathbb{Q})$ comprising the Brownian motion $\{B_t, t \geqslant 0\}$ and the exponential random variable $e \sim Exp(1)$, further assuming frictionless markets, no arbitrage and taking the equivalent martingale measure $\mathbb{Q}$ as given.

The authors assume that $\sigma(S, t)$ and $\lambda(S, t)$ remain bounded as $S \to \infty$ and so the process does not explode to infinity but, on the other hand, they do not assume that $\sigma(S, t)$ and $\lambda(S, t)$ remain bounded as $S \to 0$. This implies that the process may hit zero depending on how $\sigma(S, t)$ and $\lambda(S, t)$ behave. In general, default can happen at time τ_0 via diffusion to zero or at time $\tilde{\zeta}$ via jump to default, whichever comes first.

The time of default ζ can then be decomposed into a predictable and a totally inaccessible part given by

$$\zeta = \tau_0 \wedge \tilde{\zeta}, \tag{4.2}$$

where, for the first part, bankruptcy occurs at the first passage time of the stock price to 0

$$\tau_0 := \inf\{t > t_0 : S_t = 0\}, \tag{4.3}$$

and, for the second part, the stock price can jump to default at the first jump time

$$\tilde{\zeta} := \inf\left\{ t > t_0 : \frac{1}{\mathbb{1}_{\{t<\tau_0\}}} \int_{t_0}^{t} \lambda(u, S)\, du \geq \Theta \right\}, \tag{4.4}$$

of the integrated hazard process to the level drawn from an exponential random variable Θ independent of $W_t^{\mathbb{Q}}$ and with unit mean. Following R. J. Elliott *et al.* (2000), $\mathbb{D} = \{\mathcal{D}_t, t \geq t_0\}$ is the filtration generated by the default indicator process $\mathcal{D}_t = \mathbb{1}_{\{t>\zeta\}}$.

In accordance with Cox (1975), Carr and Linetsky (2006) account for the leverage effect and the implied volatility skew by specifying the instantaneous stock volatility as a power function[2]

$$\sigma(t, S) = a_t S_t^{\bar{\beta}}, \tag{4.5}$$

where $\bar{\beta} < 0$ represents the volatility elasticity parameter and $a_t > 0$, $\forall t$ is the time-dependent volatility scale parameter. The authors further assume consistency with the empirical evidence linking corporate bond yields and CDS spreads to equity volatility by specifying that the default intensity is an affine function of the instantaneous variance of the underlying stock

$$\lambda(t, S) = b_t + c\,\sigma(t, S)^2, \tag{4.6}$$

where $c \geq 0$ is a positive constant parameter governing the sensitivity of λ to σ^2, and $b_t \geq 0$, $\forall t$, is a deterministic non-negative function of time.

Under the unified modeling framework of Carr and Linetsky (2006), taking $\mathcal{G}_t = \mathcal{F}_t \vee \mathcal{D}_t$, and assuming no default occurring by time t_0 (i.e. $\zeta > t_0$), then the time-t_0 value of a European-style call (if $\phi = -1$) or put (if $\phi = 1$) on the stock price S, with strike K, recovery value R, and maturity date T ($\geq t_0$), can be represented by the following building blocks

$$v_{t_0}(S_{t_0}, K, T, R; \phi, \eta) = v_{t_0}^0(S_{t_0}, K, T; \phi) + v_{t_0}^D(S_{t_0}, R, T; \phi, \eta), \tag{4.7}$$

where

$$v_{t_0}^0(S_{t_0}, K, T; \phi) := \mathbb{E}_{\mathbb{Q}}\left[e^{-\int_{t_0}^T r_l dl} (\phi K - \phi S_T)^+ \mathbb{1}_{\{\zeta > T\}} \middle| \mathcal{G}_{t_0} \right], \tag{4.8}$$

is the option value but conditional on no default by time T, and

$$v_{t_0}^D(S_{t_0}, R, T; \phi, \eta) := \mathbb{E}_{\mathbb{Q}}\left[e^{-\int_{t_0}^\eta r_l dl} (\phi R)^+ \mathbb{1}_{\{\zeta \leq T\}} \middle| \mathcal{G}_{t_0} \right], \tag{4.9}$$

for $\eta \in \{\zeta, T\}$. The recovery claims with $\eta = T$ correspond to defaultable zero-coupon bonds under *fractional recovery of treasury* and with $\eta = \zeta$ correspond to defaultable zero-coupon bonds under *fractional recovery of face value* — see, for instance, Lando (2009, p. 120). For the case of an European call, there is no recovery if the firm defaults. However, for the European put, equation (4.9) corresponds to a recovery payment equal to the strike (i.e. $R = K$), that can be paid at the default time ζ or at the maturity date T, depending on the recovery assumption.

[2] We use notation consistent with Ruas *et al.* (2013).

4.2 JDCEV pricing solutions to European-style options

Assuming that $\zeta > t_0$, and constant r, q, a, b, and c, Carr and Linetsky (2006, Prop. 5.5) show that the t_0-price of an European-style call option with strike price K and expiry date at time T $(\geq t_0)$ is given by

$$\begin{aligned} c_{t_0}(S,K,T) &= e^{-q(T-t_0)}\, S\, \Phi_{+1}\left(0, \frac{k^2}{\tau}; \delta_+, \frac{x^2}{\tau}\right) \\ &\quad - e^{-(r+b)(T-t_0)}\, K \left(\frac{x^2}{\tau}\right)^{\frac{1}{2|\bar{\beta}|}} \Phi_{+1}\left(-\frac{1}{2|\bar{\beta}|}, \frac{k^2}{\tau}; \delta_+, \frac{x^2}{\tau}\right), \end{aligned} \tag{4.10}$$

and the t_0-price of the European-style put, conditional on no default by time T, is given by

$$\begin{aligned} p^0_{t_0}(S,K,T) &= e^{-(r+b)(T-t_0)}\, K \left(\frac{x^2}{\tau}\right)^{\frac{1}{2|\bar{\beta}|}} \Phi_{-1}\left(-\frac{1}{2|\bar{\beta}|}, \frac{k^2}{\tau}; \delta_+, \frac{x^2}{\tau}\right) \\ &\quad - e^{-q(T-t_0)}\, S_{t_0}\, \Phi_{-1}\left(0, \frac{k^2}{\tau}; \delta_+, \frac{x^2}{\tau}\right), \end{aligned} \tag{4.11}$$

where

$$x := \frac{1}{|\beta|} S_{t_0}^{|\beta|}, \tag{4.12}$$

$$k := \frac{1}{|\beta|} K^{|\bar{\beta}|} e^{-|\beta|(r-q+b)(T-t_0)}, \tag{4.13}$$

$$\delta_+ := \frac{2c+1}{|\beta|} + 2, \tag{4.14}$$

and

$$\tau := \begin{cases} a^2(T-t_0) & \Leftarrow \quad r-q+b=0 \\ \frac{a^2}{2|\beta|(r-q+b)}\left(1-e^{-2|\beta|(r-q+b)(T-t_0)}\right) & \Leftarrow \quad r-q+b \neq 0 \end{cases}. \tag{4.15}$$

The functions $\Phi_\theta(p,y;v,\lambda) := \mathbb{E}^{\chi^2(v,\lambda)}\left(X^p \mathbb{1}_{\{\theta X \geq \theta y\}}\right)$, for $\theta \in \{-1,1\}$, are defined by Carr and Linetsky (2006, Eq. 5.11 and 5.12) to be the truncated p-th moments of a non-central chi-square random variable X with v degrees of freedom and non-centrality parameter λ.

For the European-style put option, the recovery part at time-t_0 to be paid at maturity date T, is given by

$$p^D_{t_0}(S,K,T) = Ke^{-r(T-t_0)}\left(1-Q(S,t_0;T)\right), \tag{4.16}$$

where,

$$Q(S,t_0;T) = e^{-b(T-t_0)} \left(\frac{x^2}{\tau}\right)^{\frac{1}{2|\beta|}} M\left(-\frac{1}{2|\beta|};\delta_+,\frac{x^2}{\tau}\right), \tag{4.17}$$

represents the risk-neutral survival probability, and $M(p;n,\lambda) := \mathbb{E}^{\chi^2(n,\lambda)}(X^p)$ is the p-th raw moment of a non-central chi-square random variable X with n degrees of freedom and non-centrality parameter λ, as defined in Carr and Linetsky (2006, Eq. 5.10). Following equations (4.11) and (4.16), the t_0-price of an European-style put option is given by

$$p_{t_0}(S,K,T) = p^0_{t_0}(S,K,T) + p^D_{t_0}(S,K,T). \tag{4.18}$$

For the put option contracts paying also the value R, but at default time ζ (i.e. considering the fractional recovery of face value assumption), following Carr and Linetsky (2006, Eq. 5.15), the value of a claim that pays R dollars at the default time ζ is given by

$$\begin{aligned} p^D_{t_0}(S_{t_0},K,T) = &R\int_{t_0}^{T} e^{-(r+b)(u-t_0)}\left[b\left(\frac{x^2(S_{t_0})}{\tau(t_o,u)}\right)^{\frac{1}{2|\bar{\beta}|}} M\left(-\frac{1}{2|\bar{\beta}|};\delta_+,\frac{x^2(S_{t_0})}{\tau(t_o,u)}\right)\right.\\ &+ ca^2 S_{t_0}^{2\bar{\beta}} e^{-2|\bar{\beta}|(r-q+b)(u-t_0)}\left(\frac{x^2(S_{t_0})}{\tau(t_o,u)}\right)^{\frac{1}{2|\bar{\beta}|}+1}\\ &\left. M\left(-\frac{1}{2|\bar{\beta}|}-1;\delta_+,\frac{x^2(S_{t_0})}{\tau(t_o,u)}\right)\right]du. \end{aligned} \tag{4.19}$$

Chapter 5

Algorithms

The CIR and the CEV option pricing models make use of the non-central chi-square distribution function.

Extensive research has been devoted to the efficient computation of this distribution function — see, for instance, Farebrother (1987), Posten (1989), Schroder (1989), Ding (1992), Knüsel and Bablok (1996), Benton and Krishnamoorthy (2003) and Dyrting (2004).

A comprehensive overview of alternative methods to compute the complementary non-central chi-square distribution function is provided in Larguinho *et al.* (2013). These authors make comparisons for performance, in terms of accuracy and computational burden, for the alternative methods to compute such kind of probability distributions in the context of CEV option prices and Greeks. They find that the Gamma series method and the iterative procedures provided by Schroder (1989), Ding (1992) and Benton and Krishnamoorthy (2003) are accurate for a wide scope of parameters but present significant differences in computation speeds. Additionally, they find that the analytic approximations of Sankaran (1963), Fraser *et al.* (1998) and Penev and Raykov (2000) are fast, but when ω and λ are small they produce significant errors. They conclude by pointing out that the computer experiments performed evidence that the Benton and Krishnamoorthy (2003) clearly offers the best speed-accuracy tradeoff.

Benton and Krishnamoorthy (2003) offer an accurate and efficient way to compute the non-central chi-square distribution function. Following, for instance, Abramowitz and Stegun (1972, p. 26.4.25), the authors provide the cumulative distribution function of a non-central chi-square random variable as a series solution encompassing Poisson probabilities and the incomplete Gamma function. In Benton and Krishnamoorthy (2003, Eqs. 4.3 and 4.4), the efficiency of the algorithm is enhanced by evaluating the incomplete Gamma functions using the recurrence relations offered by Abramowitz and Stegun (1972, Eqs. 6.5.21 and 6.5.23). To save time and mitigate underflow errors, the series is initiated at the integer closest to the mean of the Poisson distribution. The algorithm finally truncates the series when the sum of the remaining error of the series is below some pre-specified error tolerance.

The JDCEV option pricing model makes use of the non-trivial evaluation of the raw and truncated moments from a non-central chi-square distribution function.

Marchand (1996, Lemma 2) provides an explicit solution to compute the truncated moments of the non-central chi-square cumulative density function. Nevertheless, the elegant solution provided by the authors is only valid for moments of integer order.

Gil *et al.* (2013) provide an effective path to compute the moments of real order for the partial non-central chi-square distribution function relying on a recurrence relation based on Bessel function ratios that minimizes overflow errors.

Dias and Nunes (2014) propose a fast and accurate algorithm to compute the truncated moments of a non-central chi-square random variable. Their method relies on forward and backward relations for the incomplete Gamma function. They apply it in the pricing of financial options under JDCEV. Their algorithm is an extension of Benton and Krishnamoorthy (2003, Algorithm 7.3). Following Carr and Linetsky (2006, Eqs. 5.11 and 5.12), the authors develop a series solution for the truncated moments of the non-central chi-square distribution, involving Poisson probabilities and the incomplete Gamma function. But, this time, the real order of the moment will be present as the first argument of the incomplete Gamma function, and so, more general recursions than those offered in Abramowitz and Stegun (1972, Eqs. 6.5.21 and 6.5.23) are needed. The proposed algorithm is similar to Benton and Krishnamoorthy (2003, Algorithm 7.3) and encompasses Benton and Krishnamoorthy (2003, Eqs. 4.3 and 4.4) as a particular case. The authors solutions are tested against Marchand (1996, Lemma 2) explicit solution. The authors also note that since any raw moment can be stated as the sum of two truncated moments — see, for instance, Carr and Linetsky (2006, Eq. 5.13) — the proposed algorithm can also be applied to the evaluation of raw moments from a non-central chi-square law. The results of the numerical analysis highlight the robustness of the algorithm that is shown to provide better speed-accuracy than the usual resource to evaluate the Kummer confluent hypergeometric function as stated, for instance, in Carr and Linetsky (2006, Eq. 5.10).

In the following sections, after describing the non-central chi-square distribution, we will review the works of Gil *et al.* (2012), Gil *et al.* (2013), Gil *et al.* (2014), Gil *et al.* (2015), Sun *et al.* (2010) and Dias and Nunes (2014), as they constitute building blocks in the search for the improvement of the efficiency of CEV, CIR and JDCEV option pricing models.

5.1 Non-central chi-square distribution

The non-central chi-square distribution was first obtained by R. A. Fisher (1928, p. 663), as a limiting case of the distribution of the multiple correlation coefficients. Being very close to the normal distribution, the non-central chi-square distribution appears frequently in finance, estimation theory and in time series analysis — see, for instance, Scharf and Demeure (1991). As described in Dyrting (2004), despite being a well known function, the non-central chi-square distribution function is sometimes difficult to evaluate accurately and efficiently, in part due to its multiple arguments. Whereas most special functions have one or to arguments, the non-central chi-square distribution has three: the number of degrees of freedom, the non-centrality parameter and the distribution's boundary.

Being Z_1, Z_2,..., Z_υ independent unit normal random variables, and δ_1, δ_2, ..., δ_υ constants, then

$$Y = \sum_{j=1}^{\upsilon} (Z_j + \delta_j)^2 \,, \tag{5.1}$$

where Y is the non-central chi-square distribution with υ degrees of freedom and non-centrality parameter $\lambda = \sum_{j=1}^{\upsilon} \delta_j^2$. In the case of $\lambda = 0$, which implies that all δ will be zero, the distribution Y will be a central chi-square distribution with υ degrees of freedom and we denote it by χ_υ^2.

Hereafter, we define $p_{\chi_\upsilon^2(\lambda)}(\omega) = p(\omega; \upsilon, \lambda)$ as the probability density function of a non-central chi-square distribution $\chi_\upsilon^2(\lambda)$ and $p_{\chi_\upsilon^2}(\omega) = p(\omega; \upsilon, 0)$ as the probability density function of a central chi-square distribution χ_υ^2. Furthermore, $P\left[\chi_\upsilon^2(\lambda) \leqslant \omega\right] = F(\omega; \upsilon, \lambda)$ represents the cumulative distribution function of $\chi_\upsilon^2(\lambda)$ and $P\left[\chi_\upsilon^2 \leqslant \omega\right] = F(\omega; \upsilon, 0)$ represents the cumulative distribution function of χ_υ^2. The notations $Q(\omega; \upsilon, \lambda)$ and $Q(\omega; \upsilon, 0)$ stand for the complementary distribution functions of $\chi_\upsilon^2(\lambda)$ and χ_υ^2, respectively.

One of several available representations of the cumulative distribution function of χ_υ^2

(see, for instance, Johnson *et al.* (1995, Eq. 29.2)) is given by

$$
\begin{aligned}
P\left[\chi^2_v(\lambda) \leqslant \omega\right] &= F(\omega; v, \lambda) \\
&= e^{-\lambda/2} \sum_{j=0}^{\infty} \frac{(\lambda/2)^j}{j! 2^{v/2+j} \Gamma(v/2+j)} \\
&\quad \times \int_0^{\omega} y^{v/2+j-1} e^{-y/2} \mathrm{d}y, \quad \omega > 0,
\end{aligned}
\tag{5.2}
$$

where

$$
\Gamma(a) := \int_0^{\infty} t^{a-1} e^{-t} dt, a > 0, \tag{5.3}
$$

represents the Euler Gamma function as defined in Abramowitz and Stegun (1972, Eq. 6.1.1), while $F(\omega; v, \lambda) = 0$ for $\omega < 0$.

It is also possible to express $F(\omega; v, \lambda)$ for $\omega > 0$ as a weighted sum of central chi-square probabilities where weights are equal to the probabilities of a Poisson distribution, where the Poisson parameter is one-half of the non-centrality parameter of the non-central chi-square distribution (see, for instance, Johnson *et al.* (1995, Eq. 29.3), or Abramowitz and Stegun (1972, Eq. 26.4.25))

$$
\begin{aligned}
F(\omega; v, \lambda) &= \sum_{j=0}^{\infty} \left(\frac{(\lambda/2)^j}{j!} e^{-\lambda/2} \right) P\left[\chi^2_{v+2j} \leqslant \omega\right] \\
&= \sum_{j=0}^{\infty} \left(\frac{(\lambda/2)^j}{j!} e^{-\lambda/2} \right) F(\omega; v+2j, 0) \\
&= \sum_{j=0}^{\infty} \left(\frac{(\lambda/2)^j}{j!} e^{-\lambda/2} \right) \frac{\gamma(\frac{v}{2}+j, \frac{\omega}{2})}{\Gamma(\frac{v}{2}+j)},
\end{aligned}
\tag{5.4}
$$

where $F(w; v{+}2, 0)$ represents the central chi-square probability function as given in Abramowitz and Stegun (1972, Eq. 26.4.), λ the non-centrality parameter, and

$$
\gamma(a, x) := \int_0^{x} t^{a-1} e^{-t} dt, a > 0, \tag{5.5}
$$

and

$$
\Gamma(a, x) := \int_x^{\infty} t^{a-1} e^{-t} dt, a > 0, \tag{5.6}
$$

being, respectively, the *lower* incomplete Gamma function and the *upper* incomplete Gamma function as defined in Abramowitz and Stegun (1972, Eqs. 6.5.2 and 6.5.3).

The complementary distribution function of the non-central chi-square function $\chi_\upsilon^2(\lambda)$ is given by

$$
\begin{aligned}
Q(\omega; \upsilon, \lambda) &= 1 - F(\omega; \upsilon, \lambda) \\
&= \sum_{j=0}^{\infty} \left(\frac{(\lambda/2)^j}{j!} e^{-\lambda/2} \right) Q(\omega; \upsilon + 2j, 0) \\
&= \sum_{j=0}^{\infty} \left(\frac{(\lambda/2)^j}{j!} e^{-\lambda/2} \right) \frac{\Gamma(\frac{\upsilon}{2} + j, \frac{\omega}{2})}{\Gamma(\frac{\upsilon}{2} + j)},
\end{aligned} \tag{5.7}
$$

where the complementary central chi-square probability function $Q(\omega; \upsilon+2j, 0)$ is as defined in Abramowitz and Stegun (1972, Eq. 26.4.2).

The probability density function of the non-central chi-square function $\chi_\upsilon^2(\lambda)$ can also be defined as a mixture of central chi-square probability density functions (see, for instance, Johnson *et al.* (1995, Eq. 29.4) or Benton and Krishnamoorthy (2003, Eq. 4.1))

$$
\begin{aligned}
p_{\chi_\upsilon^2(\lambda)}(\omega) &:= \frac{1}{2} e^{-\frac{\lambda+u}{2}} \left(\frac{u}{\lambda} \right)^{\frac{\upsilon}{4}-\frac{1}{2}} I_{\frac{\upsilon}{2}-1}\left(\sqrt{\lambda u} \right) \\
&= \sum_{j=0}^{\infty} \frac{e^{-\frac{\lambda}{2}} \left(\frac{\lambda}{2}\right)^j}{j!} \frac{e^{-\frac{x}{2}} x^{\frac{\upsilon}{2}+j-1}}{2^{\frac{\upsilon}{2}+j} \Gamma\left(\frac{\upsilon}{2}+j\right)},
\end{aligned} \tag{5.8}
$$

where $I_q(.)$ represents the *modified Bessel function* of the first kind of order q, as given in Abramowitz and Stegun (1972, Eq. 9.6.10)

$$
I_q(z) = \left(\frac{z}{2} \right)^q \sum_{j=0}^{\infty} \frac{(z^2/4)^j}{j! \Gamma(q+j+1)}. \tag{5.9}
$$

As shown in Larguinho *et al.* (2013, Eqs. 7 and 8), we can alternatively express the functions $F(\omega; \upsilon, \lambda)$ and $Q(\omega; \upsilon, \lambda)$ as integral representations

$$
F(\omega; \upsilon, \lambda) = \int_0^{\omega} \frac{1}{2} e^{-(\lambda+u)/2} \left(\frac{u}{\lambda} \right)^{(\upsilon-2)/4} I_{(\upsilon-2)/2}(\sqrt{\lambda u}) \mathrm{d}u, \tag{5.10}
$$

and

$$
Q(\omega; \upsilon, \lambda) = \int_{\omega}^{\infty} \frac{1}{2} e^{-(\lambda+u)/2} \left(\frac{u}{\lambda} \right)^{(\upsilon-2)/4} I_{(\upsilon-2)/2}(\sqrt{\lambda u}) \mathrm{d}u. \tag{5.11}
$$

5.1.1 Marcum functions and relations to the non-central chi-square function

For $\lambda \geq 0$ and $x \geq 0$, the *generalized Marcum Q-function* of real order $p \geq 0$ is defined by

$$Q_p(\lambda, x) = \frac{1}{\lambda^{p-1}} \int_x^\infty u^p e^{-\frac{u^2+\lambda^2}{2}} I_{p-1}(\lambda u)\, du, \tag{5.12}$$

where I_p is the modified Bessel function of the first kind and order p, as defined in equation (5.9).

In the case of $p = 1$, then (5.12) reduces to the *standard Marcum Q-function* $Q(a, b)$, as originally defined by Marcum (1960).

An alternative representation for the generalized Marcum *Q*-function is offered by Gil *et al.* (2014, Eq. 1)

$$\widetilde{Q}_v(\lambda, x) := \int_x^\infty e^{-(\lambda+u)} \left(\frac{u}{\lambda}\right)^{(v-1)/2} I_{v-1}(2\sqrt{\lambda u}) du, \tag{5.13}$$

and where $\lambda, x \geq 0$ and $v > 0$.

As shown by Gil *et al.* (2014, Eq. 5), the relation between the alternative representations in equations (5.12) and (5.13) is defined by

$$\widetilde{Q}_v(\lambda, x) = Q_v(\sqrt{2\lambda}, \sqrt{2x}). \tag{5.14}$$

The generalized Marcum Q-function as defined in (5.13) and its complementary

$$\widetilde{P}_v(\lambda, x) := \int_0^x e^{-(\lambda+u)} \left(\frac{u}{\lambda}\right)^{(v-1)/2} I_{v-1}(2\sqrt{\lambda u}) du, \tag{5.15}$$

satisfy the relation $\widetilde{P}_v(\lambda, x) + \widetilde{Q}_v(\lambda, x) = 1$ and they yield the non-central chi-square cumulative and complementary distribution functions as defined in equations (5.10) and (5.11). It can be shown that

$$Q(2x; 2v, 2\lambda) = \widetilde{Q}_v(\lambda, x) = Q_v(\sqrt{2\lambda}, \sqrt{2x}), \tag{5.16}$$

and

$$F(2x; 2v, 2\lambda) = \widetilde{P}_v(\lambda, x) = P_v(\sqrt{2\lambda}, \sqrt{2x}). \tag{5.17}$$

5.1.2 Nuttall functions and relations to the non-central chi-square function

The *standard Nuttall Q-function* is a generalization of the Marcum Q-function, and was initially defined by Nuttall (1972, Eq. 86),

$$Q_{p,v}(\lambda, x) := \int_x^\infty u^p e^{-\left(u^2+\lambda^2\right)/2} I_v\left(\lambda u\right) du, \tag{5.18}$$

where $x, p, v \geq 0$, $\lambda \geq 0$, and where I_v is the modified Bessel function of the first kind and order v, as defined in equation (5.9).

Kapinas *et al.* (2009) and Sun *et al.* (2010) alternatively define a *normalized Nuttall Q-function* as

$$\mathcal{Q}_{p,v}(\lambda, x) := \frac{Q_{p,v}(\lambda, x)}{\lambda^v}, \tag{5.19}$$

that when $p = v + 1$ reduces to the generalized Marcum Q-function of order $v + 1$

$$\mathcal{Q}_{v+1,v}(\lambda, x) = Q_{v+1}(\lambda, x), \tag{5.20}$$

for all admissible values of v, λ and x.

A different representation is proposed for the Nuttall Q-function by Gil *et al.* (2013, Eq. 4) and Ruas *et al.* (2013, Eq. D.4)

$$\widetilde{Q}_{p,v}(\lambda, x) := \lambda^{\frac{1-v}{2}} \int_x^\infty u^{p+\frac{v-1}{2}} e^{-u-\lambda} I_{v-1}\left(2\sqrt{\lambda u}\right) du, \tag{5.21}$$

which, following Ruas *et al.* (2013, Eq. D.2), can be related to the standard Nuttall Q-function defined in (5.18) by

$$\widetilde{Q}_{p,v}(\lambda, x) = 2^{-p} \left(2\lambda\right)^{(1-v)/2} Q_{2p+v,v-1}\left(\sqrt{2\lambda}, \sqrt{2x}\right). \tag{5.22}$$

The Nuttall Q-function representations offered in equations (5.18), (5.19) and (5.21), can be related to is (complementary) Nuttall P-function by changing the range of integration[1]

$$\widetilde{P}_{p,v}(\lambda, x) := \lambda^{\frac{1-v}{2}} \int_0^x u^{p+\frac{v-1}{2}} e^{-u-\lambda} I_{v-1}\left(2\sqrt{\lambda u}\right) du, \tag{5.23}$$

[1]For the Nuttall P-function $\widetilde{P}_{p,v}(\lambda, x)$ relation with its complementary $\widetilde{Q}_{p,v}(\lambda, x)$, an identity if offered in Ruas *et al.* (2013, Eq. D.3).

The p-th raw moment and the *lower tail* and *upper tail* of the truncated moment of a random variable $X \sim \chi^2(v, \lambda)$ can be defined as (see, for instance, Carr and Linetsky (2006, Lemma 5.1))

$$\begin{aligned} M(p; v, \lambda) &= \mathbb{E}^{\chi^2(v,\lambda)}\left(X^p\right) \\ &= 2^p e^{-\frac{\lambda}{2}} \frac{\Gamma(p+\frac{v}{2})}{\Gamma(\frac{v}{2})} {}_1F_1\left(p+\frac{v}{2}, \frac{v}{2}, \frac{\lambda}{2}\right), \end{aligned} \tag{5.24}$$

$$\begin{aligned} \Phi_{+\theta}\left(p, x; v, \lambda\right) &:= \mathbb{E}^{\chi^2(v,\lambda)}\left(X^p \mathbb{1}_{\{X>x\}}\right) \\ &= 2^p \sum_{i=0}^{\infty} \frac{e^{-\frac{\lambda}{2}}\left(\frac{\lambda}{2}\right)^i}{i!} \frac{\Gamma(p+\frac{v}{2}+i, \frac{x}{2})}{\Gamma(\frac{v}{2}+i)}, \end{aligned} \tag{5.25}$$

$$\begin{aligned} \Phi_{-\theta}\left(p, x; v, \lambda\right) &:= \mathbb{E}^{\chi^2(v,\lambda)}\left(X^p \mathbb{1}_{\{X\leq x\}}\right) \\ &= 2^p \sum_{i=0}^{\infty} \frac{e^{-\frac{\lambda}{2}}\left(\frac{\lambda}{2}\right)^i}{i!} \frac{\gamma(p+\frac{v}{2}+i, \frac{x}{2})}{\Gamma(\frac{v}{2}+i)}, \end{aligned} \tag{5.26}$$

where $\mathbb{E}^{\chi^2(v,\lambda)}$ denotes the expectation respecting the law of a non-central chi-square random variable, with v degrees of freedom and non-centrality parameter λ, and where

$${}_1F_1(a, b, z) := \sum_{i=0}^{\infty} \frac{(a)_i}{(b)_i} \frac{z^i}{i!}, \tag{5.27}$$

is the *Kummer confluent hypergeometric function of the first kind* as defined, for instance, in Abramowitz and Stegun (1972, Eq. 13.1.2) and $(a)_i$ is the *Pochhamer* function as defined in Abramowitz and Stegun (1972, p. 6.1.22).

By definition, the three functions defined in equations (5.24) to (5.26) satisfy the identity

$$\Phi_{+\theta}\left(p, x; v, \lambda\right) + \Phi_{-\theta}\left(p, x; v, \lambda\right) = M(p; v, \lambda), \tag{5.28}$$

for any $x > 0$.

Recently, Dias and Nunes (2016) presented a series solution for p-th moment about zero of a random variable $X \sim \chi^2(v, \lambda)$ defined as

$$\begin{aligned} M(p; v, \lambda) &= 2^p \sum_{i=0}^{\infty} \frac{e^{-\frac{\lambda}{2}} \left(\frac{\lambda}{2}\right)^i}{i!} \frac{\Gamma\left(p + \frac{v}{2} + i\right)}{\Gamma\left(\frac{v}{2} + i\right)} \\ &= 2^p \sum_{i=0}^{\infty} P_i \widetilde{I}\left(\frac{v}{2} + i, p\right), \end{aligned} \tag{5.29}$$

where P_i is the Poisson density (with mean $\frac{\lambda}{2}$)

$$P_i := \frac{e^{-\frac{\lambda}{2}} \left(\frac{\lambda}{2}\right)^i}{i!}. \tag{5.30}$$

The truncated moments in the first step of equations (5.25) and (5.26), can be related to the Nuttall function representations of equations (5.18) and (5.21) by

$$\Phi_{+\theta}\left(p, 2x; 2v, 2\lambda\right) = 2^p \widetilde{Q}_{p,v}(\lambda, x) = (2\lambda)^{(1-v)/2} Q_{2p+v,v-1}\left(\sqrt{2\lambda}, \sqrt{2x}\right), \tag{5.31}$$

and, as shown by Ruas *et al.* (2013, Eq. D.1), to the representation of equation (5.23) by

$$\Phi_{-\theta}\left(p, 2x; 2v, 2\lambda\right) = 2^p \widetilde{P}_{p,v}(\lambda, x). \tag{5.32}$$

5.1.3 Auxiliary derivations

As in Dias and Nunes (2014), to save space and using the incomplete Gamma function ratios $P(a, x) = \gamma(a, x)/\Gamma(a)$ and $Q(a, x) = \Gamma(a, x)/\Gamma(a)$ as shown in Abramowitz and Stegun (1972, Eq. 26.4.19), the following definition will be adopted hereafter for $\theta \in \{-1, 1\}$:

$$I(a, x, p; \theta) := \begin{cases} \frac{\gamma(a+p,x)}{\Gamma(a)} = \frac{\Gamma(a+p)}{\Gamma(a)} P(a+p, x) \Leftarrow \theta = -1 \\ \frac{\Gamma(a+p,x)}{\Gamma(a)} = \frac{\Gamma(a+p)}{\Gamma(a)} Q(a+p, x) \Leftarrow \theta = 1 \end{cases}, \tag{5.33}$$

Definition (5.33) allows the series solutions (5.25) and (5.26) to be summarized as

$$\Phi_\theta(p, x; n, \lambda) = 2^p \sum_{i=0}^{\infty} P_i I\left(\frac{n}{2} + i, \frac{x}{2}, p; \theta\right), \tag{5.34}$$

where

$$P_i := \frac{e^{-\frac{\lambda}{2}} \left(\frac{\lambda}{2}\right)^i}{i!}, \tag{5.35}$$

is a Poisson density with mean $\lambda/2$.

Therefore

$$\Phi'_{\theta}(p,x;n,\lambda) := \frac{\partial}{\partial\lambda}\Phi_{\theta}(p,x;n,\lambda) = 2^p \sum_{i=0}^{\infty} \left(\frac{i}{\lambda} - \frac{1}{2}\right) P_i I\left(\frac{n}{2}+i, \frac{x}{2}, p; \theta\right). \tag{5.36}$$

The p-th moment about zero of a random variable $X \sim \chi^2(v,\lambda)$ can be defined as a series solution by

$$\begin{aligned} M(p;v,\lambda) &= 2^p \sum_{i=0}^{\infty} \frac{e^{-\frac{\lambda}{2}}\left(\frac{\lambda}{2}\right)^i}{i!} \frac{\Gamma\left(p+\frac{v}{2}+i\right)}{\Gamma\left(\frac{v}{2}+i\right)} \\ &= 2^p \sum_{i=0}^{\infty} P_i \widetilde{I}\left(\frac{v}{2}+i, p\right), \end{aligned} \tag{5.37}$$

where P_i is the Poisson density (with mean $\frac{\lambda}{2}$) given in equation (5.35).

Therefore

$$M'(p;v,\lambda) = \frac{\partial}{\partial\lambda} M(p;v,\lambda) = 2^p \sum_{i=0}^{\infty} \left(\frac{i}{\lambda} - \frac{1}{2}\right) P_i \widetilde{I}\left(\frac{v}{2}+i, p\right). \tag{5.38}$$

5.2 Incomplete Gamma function ratios by Gil et al. (2012)

For the option pricing models we are addressing, the incomplete Gamma function is of prime relevance. It is employed in the computation of the non-central chi-square distribution function used in the CEV and CIR models and also in the truncated and raw moments for the computation of the JDCEV option pricing model.

In this paper, the authors present numerical algorithms to evaluate the incomplete Gamma functions ratios $P(a,x) = \gamma(a,x)/\Gamma(a)$ and $Q(a,x) = \Gamma(a,x)/\Gamma(a)$ for positive values of a and x. The authors also present inversion methods for solving $P(a,x) = p$ and $Q(a,x) = q$, with $0 < p$ and $q < 1$. The authors present a software associated with the discussed algorithms (a Fortran 90 module called **IncgamFI**) and its performance is compared with earlier published algorithms – see, for instance, DiDonato and Morris (1986).

In the numerical algorithms described by the authors, both $P(a,x)$ and $Q(a,x)$ are computed. Because $P(a,x)+Q(a,x) = 1$, only one function needs to be computed, usually, the smaller of the two. For large values of a, x, the authors consider a transition at $a \sim x$, with $P(a,x) \lesssim \frac{1}{2}$ when $a \gtrsim x$ and $Q(a,x) \lesssim \frac{1}{2}$ when $a \lesssim x$. Following this, the methods of computation are divided in two zones, comprising each one several methods of computation.

The authors describe the methods that should be used for each (x, a) quarter plane. First they define a function with the purpose to separate the (x, a) quarter plane into two regions. Each region defines the primary function $P(a, x)$ and $Q(a, x)$ that should be computed first. Following Gautschi (1979), the authors define

$$\alpha(x) = \begin{cases} x & \text{if} \quad x \geqslant \frac{1}{2} \\ \frac{ln\frac{1}{2}}{ln(\frac{1}{2}x)} & \text{if} \quad 0 < x < \frac{1}{2} \end{cases}. \tag{5.39}$$

Then, the primary function can be defined as

$$\begin{array}{lll} P(a, x) & \text{when} & a \geqslant \alpha(x), \\ Q(a, x) & \text{when} & a < \alpha(x). \end{array} \tag{5.40}$$

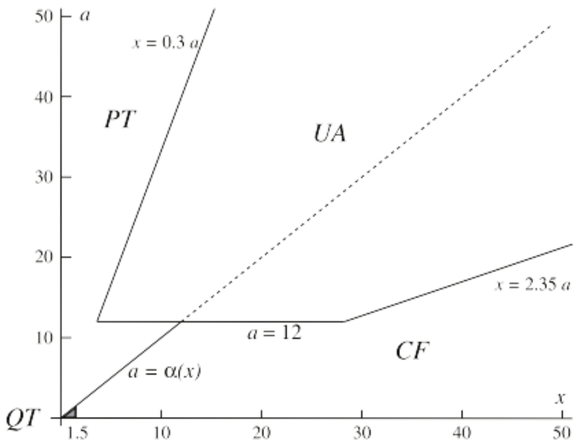


Figure 5.1: Domains for computing $P(a, x)$ and $Q(a, x)$ PT: using the Taylor expansion of $P(a, x)$ – see Gil *et al.* (2012, Section 2.2); QT: using the Taylor expansion of $Q(a, x)$ – see Gil *et al.* (2012, Section 2.3); CF using the continued fraction for $Q(a, x)$ – see Gil *et al.* (2012, Section 2.4); UA: using the uniform asymptotic methods for $P(a, x)$ and $Q(a, x)$ – Gil *et al.* (2012, Section 2.5).

The authors establish the domains based on a compromise between efficiency and accuracy, being the efficiency the prevailing factor whenever the accuracy of two methods is the same.

5.3 Moments of the partial non-central chi-square distribution function by Gil et al. (2013)

As we have seen previously, the truncated and raw moments of the non-central chi-square distribution function, play a central role in the computation of option prices under the JDCEV pricing model.

In this paper, the authors present and discuss the properties and methods of computation of the moments of the partial non-central chi-square distribution, also known as Nuttall Q-functions.

The partial non-central chi-square distribution ηth moment is given by

$$Q_{\eta,\mu}(x,y) = x^{\frac{1}{2}(1-\mu)} \int_y^{+\infty} t^{\eta+\frac{1}{2}(\mu-1)} \mathrm{e}^{-t-x} I_{\mu-1}(2\sqrt{xt})\mathrm{d}t. \tag{5.41}$$

5.3.1 Properties

The series expansion for the ηth moment of the non-central chi-square distribution function can be represented by

$$Q_{\eta,\mu}(x,y) = \mathrm{e}^{-x} \sum_{n=0}^{\infty} \frac{x^n}{n!} \frac{\Gamma(\eta+\mu+n,y)}{\Gamma(\mu+n)}, \tag{5.42}$$

that when given in terms of the incomplete Gamma function ratio[2] $Q_\mu(x) = \Gamma(\mu,x)/\Gamma(\mu)$ takes the form

$$Q_{\eta,\mu}(x,y) = \mathrm{e}^{-x} \sum_{n=0}^{\infty} \frac{x^n}{n!} \frac{\Gamma(\eta+\mu+n)}{\Gamma(\mu+n)} Q_{\eta+\mu+n(y)}. \tag{5.43}$$

By considering integration by parts, the integral in (5.41) together with the relation $z^\mu I_{\mu-1}(z) = \frac{d}{dz}(z^\mu I_\mu(z))$, a recurrence relation for the moments of the non-central chi-square distribution function, can be represented as

$$Q_{\eta,\mu}(x,y) = Q_{\eta,\mu+1}(x,y) - \eta Q_{\eta-1,\mu+1} - \left(\frac{y}{x}\right)^{\frac{\mu}{2}} y^\eta \mathrm{e}^{-x-y} I_\mu(2\sqrt{xy}). \tag{5.44}$$

This recurrence, which reduces to a first order difference equation for the Marcum Q-function when $\eta = 0$, can be used for testing and also for computation.

5.3.2 Computing moments using the series expansion

The authors test the series expansion written in (5.42) with the recurrence relation of (5.44) by writing

$$\frac{Q_{\eta,\mu+1}(x,y)}{Q_{\eta,\mu}(x,y) + \eta Q_{\eta-1,\mu+1} - \left(\frac{y}{x}\right)^{\frac{\mu}{2}} y^\eta \mathrm{e}^{-x-y} I_\mu(2\sqrt{xy})} = 1, \tag{5.45}$$

[2] Whose algorithms are given in Gil *et al.* (2012).

where the left-hand side deviations from 1 (absolute value) in (5.45) measure the accuracy of the tested method. The authors implemented a Fortran 90 module **NuttallF** to compute the series expansion. The latter module uses the Gil *et al.* (2012) module **IncgamFI** to compute the Gamma function ratios. The authors test the series expansion for the parameter region $(\eta, \mu, x, y) \in (1, 50) \times (1, 50) \times (0, 20) \times (0, 20)$ and the tests show that with the series expansion an accuracy of 10^{-12} can be obtained. To avoid overflow problems, when $\mu + n \to \infty$, the authors use

$$\frac{\Gamma(\eta + \mu + n)}{\Gamma(\mu + n)} \sim (\mu + n)^{\eta}. \tag{5.46}$$

5.3.3 Computing moments by recursion

If we write the recurrence relation of (5.44) as

$$Q_{\eta,\mu+1}(x, y) = Q_{\eta,\mu}(x, y) + \eta Q_{\eta-1,\mu+1} + \left(\frac{y}{x}\right)^{\frac{\mu}{2}} y^{\eta} \mathrm{e}^{-x-y} I_{\mu}(2\sqrt{xy}), \tag{5.47}$$

we are in the presence of a numerically stable relation since all the right-hand side terms are positive. It is worthwhile mentioning that particular care has to be taken with the application of the inhomogeneous recurrence. Overflow/underflow errors can occur due to bad conditioning of the exponentials in the Bessel function when x and/or y becomes large. Part of this problem can be avoided if one uses the scaled Bessel function $\tilde{I}_{\nu}(x) = e^{-x} I_{\nu}(x)$. Rewriting (5.47) in terms of this function, we have

$$Q_{\eta,\mu+1}(x, y) = Q_{\eta,\mu}(x, y) + \eta Q_{\eta-1,\mu+1} + \left(\frac{y}{x}\right)^{\frac{\mu}{2}} y^{\eta} \mathrm{e}^{-(\sqrt{x}-\sqrt{y})^2} \tilde{I}_{\mu}(2\sqrt{xy}). \tag{5.48}$$

Now, if we assume that we know the moments of order zero (Marcum functions) for a sequence of real values $\mu_i, i = 1, ...N$ with $\mu_{i+1} - \mu_i = 1$, and if $Q(1, \mu)$ is also known, (5.47) can be used to compute $Q(1, \mu+1)$. That means that starting from the value of $Q(1, 1)$, we can compute $Q(1, \mu), \mu = 1, 2, ..., N$ in a stable way. In the same way, after determining $Q(1, \mu), \mu = 1, 2, ..., N$, if we know $Q(2, 1)$, we can compute $Q(2, \mu), \mu = 1, 2, ..., N$ and so on.

Considering an homogeneous equation offers an alternative way of computing the recurrences. This equation can be constructed from the inhomogeneous equation by writing

$$Q_{\eta,\mu+2} - Q_{\eta,\mu+1} - \eta Q_{\eta-1,\mu+2} = c_{\mu+1}(Q_{\eta,\mu+1} - Q_{\eta,\mu} - \eta Q_{\eta-1,\mu+1}) \tag{5.49}$$

where

$$c_{\mu+1} = \sqrt{\frac{y}{x}} \frac{I_{\mu+1}(2\sqrt{xy})}{I_{\mu}(2\sqrt{xy})}. \tag{5.50}$$

If we know $Q(\eta-1,\mu)$, $\mu = 1, 2, ..., N$ it is possible to compute $Q(\eta,\mu)$, $\mu = 1, 2, ..., N$, starting from $Q(\eta, 1)$ and $Q(\eta, 2)$ with the recurrence

$$Q_{\eta,\mu+2} = (1 + c_{\mu+1})Q_{\eta,\mu+1} - c_{\mu+1}Q_{\eta,\mu} + \eta Q_{\eta-1,\mu+2} - \eta c_{\mu+1}Q_{\eta-1,\mu+1}. \tag{5.51}$$

In this equation, because Bessel function ratios are used instead of Bessel functions themselves, overflow problems are reduced. In Gil *et al.* (2013, Table 2), the authors present the relative errors obtained when comparing the values obtained with the recurrence relation of (5.51) and the series expansion of 5.43, that can be found to have an accuracy of 10^{-14}.

5.4 GammaCHI package for the inversion and computation of the Gamma and chi-square cumulative distribution by Gil et al. (2015)

In Gil *et al.* (2015), the authors present a Fortran 90 module **GammaCHI** that, in the authors view, favour reliable and fast routines for the inversion and computation of Gamma and chi-square distribution functions.

The module provided with this work includes routines where the direct computation of the central Gamma and chi-square distribution works as well as their inversion. In what the direct computation is concerned, the algorithm computes both $P(a,x)$ and $Q(a,x)$. A note should be made to the fact that computing $Q(a,x)$ simply as $1-P(a,x)$ if $P(a,x)$ is close to 1 can lead to serious cancellation problems. Also, the inversion routine solves the equations

$$P(a,x) = p, \quad Q(a,x) = q, \quad 0 < p, q < 1, \tag{5.52}$$

for a given value of a.

The authors describe in Gil *et al.* (2013) the algorithm to compute the Gamma distribution function. Nevertheless, the authors state that some improvements to its performance are included in the version included in the **GammaCHI** package. In the package, the distribution functions are computed with the use of Taylor expansions, continued fractions or uniform asymptotic expansions in combination with high order Newton methods, as we have seen in (Section 5.2).

5.5 Tight bounds of the generalized Marcum and Nuttall *Q*-functions by Sun et al. (2010)

It is well known that precise computation of the generalized Marcum and Nuttall Q-functions are difficult because of the modified Bessel function of the first kind I_ν involved in their computation. Based on the log-concavity of these functions, Sun *et al.* (2010) propose tighter bounds than the ones suggested recently by, for instance, Kapinas *et al.* (2009) and by Li and Kam (2006).

5.5.1 Marcum *Q*-functions

Sun *et al.* (2010) refer to the closed form expressions of the generalized Marcum Q-function, $Q_\nu(a,b)$, given by Li and Kam (2006, Eq. 11), for the case when ν is an odd multiple of 0.5

$$\begin{aligned} Q_\nu(a,b) = &\frac{1}{2}\text{erfc}\left(\frac{b+a}{\sqrt{2}}\right) + \frac{1}{2}\text{erfc}\left(\frac{b-a}{\sqrt{2}}\right) \\ &+ \frac{1}{a\sqrt{2\pi}}\sum_{k=0}^{\nu-1.5}\frac{b^{2k}}{2^k}\sum_{q=0}^{k}\frac{(-1)^q(2q)!}{(k-q)!q!} \\ &\times \sum_{i=0}^{2q}\frac{1}{(ab)^{2q-i}i!}\left[(-1)^i e^{-\frac{(b-a)^2}{2}} - e^{-\frac{(b+a)^2}{2}}\right], \\ &\qquad\qquad a>0, \qquad b\geq 0, \end{aligned} \tag{5.53}$$

and to a derivation of Li and Kam (2006, Eq. 12), for the case $a = 0$

$$Q_\nu(a,b) = \text{erfc}\left(\frac{b}{\sqrt{2}}\right) + e^{-\frac{b^2}{2}}\sqrt{\frac{2}{\pi}}\sum_{k=0}^{\nu-1.5}\frac{b^{2k+1}}{(2k+1)!!}, \tag{5.54}$$

where $\text{erfc}(\cdot)$ is the complementary error function as defined in Abramowitz and Stegun (1972, Eq. 7.1.2).

Considering that equations (5.53) and (5.54) represent a closed-formula solution for $Q_\nu(a,b)$ where ν is an odd multiple of 0.5 and letting $\lfloor x \rfloor$ be the maximal integer less than or equal to x then, $\nu_1 = \lfloor \nu + 0.5 \rfloor + 0.5$ represents the minimal order that is larger than ν and an odd multiple of 0.5 and $\nu_2 = \lfloor \nu - 0.5 \rfloor + 0.5$ represents the maximal order that is less than or equal to ν and also an odd multiple of 0.5. The log concavity of $\nu \mapsto Q_\nu(a, b)$ on $[1, \infty)$, as given in Sun *et al.* (2010, Theorem 3(b)), implies one lower bound for $Q_\nu(a, b)$, given by (Sun *et al.* (2010, Eq. 52))

$$\begin{aligned} Q_\nu(a,b) &\geq Q_{\nu-LB1}(a,b) \\ &= Q_{\nu_1}(a,b)^{\nu-\nu_2} Q_{\nu_2}(a,b)^{\nu_1-\nu}, \qquad \nu \geq 1.5, \end{aligned} \tag{5.55}$$

and two upper bounds for $Q_\nu(a, b)$, given by (Sun *et al.* (2010, Eq. 55))

$$Q_\nu(a,b) \leq Q_{\nu-UB1}(a,b) = \frac{Q_{\nu_1}(a,b)^{\nu_1-\nu+1}}{Q_{\nu_1+1}(a,b)^{\nu_1-\nu}}, \qquad \nu \geq 1, \tag{5.56}$$

and (Sun *et al.* (2010, Eq. 56))

$$Q_\nu(a,b) \leq Q_{\nu-UB2}(a,b) = \frac{Q_{\nu_2}(a,b)^{\nu-\nu_2+1}}{Q_{\nu_2-1}(a,b)^{\nu-\nu_2}}, \qquad \nu \geq 2.5. \tag{5.57}$$

From Sun *et al.* (2010, Theorem 1), the authors obtain that $\nu \mapsto Q_\nu(a, b)$ is strictly increasing for $\nu \in (0, \infty)$. Building on this result, the authors obtain another lower bound for $Q_\nu(a, b)$, given by (Sun *et al.* (2010, Eq. 61))

$$\begin{aligned} Q_\nu(a,b) &\geq Q_{\nu-LB2}(a,b) \\ &= Q_\nu(0,b) + \left[Q_{\nu_1}(a,b) - Q_{\nu_1}(0,b)\right]^{\nu-\nu_2} \\ &\times \left[Q_{\nu_2}(a,b) - Q_{\nu_2}(0,b)\right]^{\nu_1-\nu}, \qquad \nu \geq 0.5, \end{aligned} \tag{5.58}$$

and another two upper bounds for $Q_\nu(a, b)$, given by (Sun *et al.* (2010, Eq. 62))

$$\begin{aligned} Q_\nu(a,b) &\leq Q_{\nu-UB3}(a,b) \\ &= Q_\nu(0,b) + \frac{\left[Q_{\nu_1}(a,b) - Q_{\nu_1}(0,b)\right]^{\nu_1-\nu+1}}{\left[Q_{\nu_1+1}(a,b) - Q_{\nu_1+1}(0,b)\right]^{\nu_1-\nu}}, \qquad \nu > 0, \end{aligned} \tag{5.59}$$

and (Sun *et al.* (2010, Eq. 63))

$$\begin{aligned} Q_\nu(a,b) &\leq Q_{\nu-UB4}(a,b) \\ &= Q_\nu(0,b) + \frac{\left[Q_{\nu_2}(a,b) - Q_{\nu_2}(0,b)\right]^{\nu-\nu_2+1}}{\left[Q_{\nu_2-1}(a,b) - Q_{\nu_2-1}(0,b)\right]^{\nu-\nu_2}}, \qquad \nu \geq 1.5. \end{aligned} \tag{5.60}$$

5.5.2 Nuttall *Q*-functions

Sun *et al.* (2010) refer to the closed form expressions of the standard Nuttall Q-function, $Q_{\mu,\nu}(a,b)$, given by Kapinas *et al.* (2009, Th. 1), given by

$$Q_{\mu,\nu}(a,b)=\frac{(-1)^n(2a)^{-n+\frac{1}{2}}}{\sqrt{\pi}}\sum_{k=0}^{n-1}\frac{(n-k)_{n-1}(2a)^k}{k!}\mathcal{I}^k_{m,n}(a,b), \tag{5.61}$$

where $a>0$, $b\geq 0$, $\mu\geq\nu$, $m=\mu+0.5\in\mathbb{N}$, $n=\nu+0.5\in\mathbb{N}$, and where the term $\mathcal{I}^k_{m,n}$ is given by

$$\begin{aligned}\mathcal{I}^k_{m,n}(a,b)&=(-1)^{k+1}\sum_{l=0}^{m-n+k}\binom{m-n+k}{l}2^{\frac{l-1}{2}}a^{m-n+k-l}\\&\times\left[\Gamma\left(\frac{l+1}{2}\right)+(-1)^{m-n-l-1}\Gamma\left(\frac{l+1}{2},\frac{(b+a)^2}{2}\right)\right.\\&\left.-[\operatorname{sgn}(b-a)]^{l+1}\gamma\left(\frac{l+1}{2},\frac{(b-a)^2}{2}\right)\right],\end{aligned} \tag{5.62}$$

where Γ is the Euler Gamma (Abramowitz and Stegun (1972, Eq. 6.1.1)), γ is the lower incomplete Gamma (Abramowitz and Stegun (1972, Eq. 6.5.2)), $\Gamma(\cdot,\cdot)$ the upper incomplete Gamma (Abramowitz and Stegun (1972, Eq. 6.5.3)) functions, respectively, and where $\binom{\cdot}{\cdot}$ is the binomial coefficient as defined in Abramowitz and Stegun (1972, Eq. 24.1.1 C) and $\operatorname{sgn}(\cdot)$ is the signum function.

The authors further consider the normalized Nuttall Q-function $\mathcal{Q}_{\mu,\nu}(a,b)$, for $m=\mu+0.5\in\mathbb{N}$ and $n=\nu+0.5\in\mathbb{N}$, as defined in Kapinas *et al.* (2009, Corollary 1)

$$\mathcal{Q}_{\mu,\nu}(a,b)=\frac{(-1)^n2^{-n+\frac{1}{2}}}{\sqrt{\pi}a^{2n-1}}\sum_{k=0}^{n-1}\frac{(n-k)_{n-1}(2a)^k}{k!}\mathcal{I}^k_{m,n}(a,b). \tag{5.63}$$

Building on Sun *et al.* (2010, Theorem 6) where it is obtained that the function $\nu\mapsto Q_{\mu.\nu}(a,b)$ is log-concave on $[0,\infty)$ for $\mu-\nu\geq 1$ fixed, the authors define lower and upper bounds for the standard and normalized Nuttall Q-function, $Q_{\mu,\nu}(a,b)$. When $\mu-\nu\geq 1$ is an integer and letting $\mu_1=\lfloor\mu+0.5\rfloor+0.5$ be the minimal order that is larger than μ and also an odd multiple of 0.5, and $\mu_2=\lfloor\mu-0.5\rfloor+0.5$ be the maximal order that is less than or equal to μ and is an odd multiple of 0.5, the authors define one lower bound for $Q_{\mu,\nu}(a,b)$

$$
\begin{aligned}
Q_{\mu,\nu}(a,b) &\geq Q_{\mu,\nu-LB}(a,b) \\
&= Q_{\mu_1,\nu_1}(a,b)^{\nu-\nu_2} Q_{\mu_2,\nu_2}(a,b)^{\nu_1-\nu}, \qquad \nu \geq 0.5,
\end{aligned}
\tag{5.64}
$$

and two upper bounds defined as

$$
\begin{aligned}
Q_{\mu,\nu}(a,b) &\leq Q_{\mu,\nu-UB1}(a,b) \\
&= \frac{Q_{\mu_1,\nu_1}(a,b)^{\nu_1-\nu+1}}{Q_{\mu_1+1,\nu_1+1}(a,b)^{\nu_1-\nu}}, \qquad \nu \geq 0,
\end{aligned}
\tag{5.65}
$$

and

$$
\begin{aligned}
Q_{\mu,\nu}(a,b) &\leq Q_{\mu,\nu-UB2}(a,b) \\
&= \frac{Q_{\mu_2,\nu_2}(a,b)^{\nu-\nu_2+1}}{Q_{\mu_2-1,\nu_2-1}(a,b)^{\nu-\nu_2}}, \qquad \nu \geq 1.5.
\end{aligned}
\tag{5.66}
$$

5.6 Truncated moments of a non-central chi-square random variable by Dias and Nunes (2014)

Dias and Nunes (2014) propose a fast and accurate algorithm for the computation of truncated moments (of any real order) for a non-central chi-square random variable, based on forward and backward recurrence relations for the incomplete Gamma functions. Furthermore, the authors provide relations to compute the generalized Marcum *P*-function and Marcum *Q*-function as shown in Gil *et al.* (2014, Eq. 2 and 1) and the Nuttall *P*-Function and *Q*-Function as shown in Gil *et al.* (2013).

Chapter 6

Numerical Analysis

In this section, we test the algorithms proposed by Gil *et al.* (2012), Gil *et al.* (2013), Dias and Nunes (2014) and Gil *et al.* (2015).

We start by testing Gil *et al.* (2012) incomplete Gamma function ratios against a selected benchmark.

We proceed to compare Dias and Nunes (2014) algorithm to compute the Marcum and Nuttall Q-function against the tight bounds proposed by Sun *et al.* (2010).

Finally, we test the non-central chi-square distribution function and its related functions under the option pricing models of CEV and JDCEV. Firstly, under the CEV framework, we compare for speed and accuracy the call option prices computation of 2,500 contracts using Benton and Krishnamoorthy (2003) and Gil *et al.* (2015) to compute the *cumulative density function* of a non-central chi-square function against a benchmark based on Knüsel (1986) and Knüsel and Bablok (1996) and the stopping approach therein. Lastly, and under the JDCEV framework, we compare for speed and accuracy the put option prices computation of 2,500 contracts using Gil *et al.* (2013) and Dias and Nunes (2014) to compute the *truncated moments* and *raw moments* of a non-central chi-square function against a benchmark based on Carr and Linetsky (2006, Lemma 5.1) and the stopping approach in Knüsel and Bablok (1996).

All the experiments in this section were conducted using *Fortran 90* running on *UNIX GNU* compiler (version 5.2.0) or *Matlab* (version 16a), both running on a 1.8 GHz Intel Core i5 personal computer. We truncated all the iterative procedures with an error tolerance of 1E-15.

Table 6.1: Differences in incomplete Gamma function values using Gil *et al.* (2012) method against Matlab.

Method	MaxAE	MaxRE	RMSE	MeanAE
G12 ($P(a,x)$)	3.00E-15	3.73E-13	2.70E-16	7.68E-17
G12 ($Q(a,x)$)	3.00E-15	5.64E-14	2.71E-16	2.71E-16

Summary of the comparison of 40,000 pairs of values for the ratios $P(a,x) = \gamma(a,x)/\Gamma(a)$ and $Q(a,x) = \Gamma(a,x)/\Gamma(a)$, computed for the range of parameters $(x,a) \in (0,50) \times (0,50)$.

6.1 Incomplete Gamma function ratios

In this section, we test Gil *et al.* (2012) (G12) Fortran 90 **IncgamFI** incomplete Gamma function ratios module. These have a central role in the computation of Marcum and Nuttall *Q*-function and related algorithms presented earlier and tested in the subsequent sections. As we rely on Gil *et al.* (2012) package interchangeably with Matlab **gammainc** function for the computation of these rations, we find it relevant to test it against each other.

The results in table (6.1) show that both methods agree on a double precision of 1E-15, required in the subsequent chapters.

6.2 Marcum and Nuttall functions and related truncated moments results

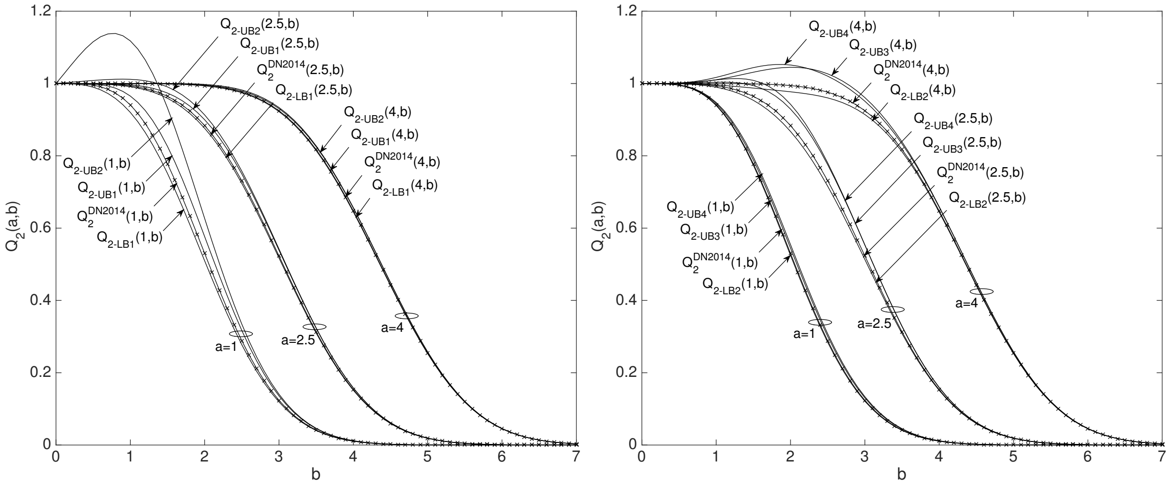
We compare the accuracy of Dias and Nunes (2014) method against the recent work of Sun *et al.* (2010) and the references therein.

Sun *et al.* (2010) provide tighter bounds for Marcum and Nuttall *Q*-functions than the ones provided in the literature[1]. The authors prove that the relative errors of the bounds converge to 0 as $b \to 0$ and provide numerical results that show that the absolute relative errors are less than 5% in most of the cases. Although the bounds proposed by the authors are proven to be quite tight, Dias and Nunes (2014) provide an algorithm to compute these functions always comprised inside these bounds.

In the following sections, we follow the combination of parameters followed by Sun *et al.* (2010) to replicate their proposed bounds and their relation to Dias and Nunes (2014) algorithm for the computation of the Marcum and Nuttall *Q*-functions[2].

[1] See, for instance, Annamalai and Tellambura (2001), Li and Kam (2006), Sun and Zhou (2008), Kapinas *et al.* (2009) and Baricz and Sun (2009).

[2] We provide the Matlab code in Appendix A.



(a) (b)

Figure 6.1: Numerical results for $Q_\nu(a,b)$ and the proposed bounds versus b for $a \in \{1, 2.5, 4\}$ and $\nu = 2$. (a) The bounds $Q_{\nu-LB1}(a,b)$, $Q_\nu^{DN2014}(a,b)$, $Q_{\nu-UB1}(a,b)$ and $Q_{\nu-UB2}(a,b)$. The bounds $Q_{\nu-LB2}(a,b)$, $Q_\nu^{DN2014}(a,b)$, $Q_{\nu-UB3}(a,b)$ and $Q_{\nu-UB4}(a,b)$

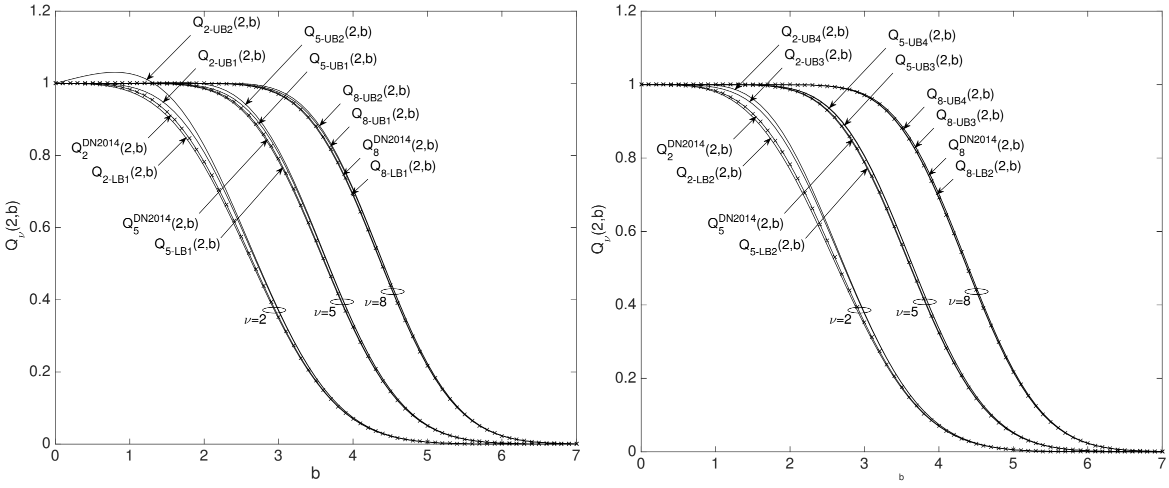
6.2.1 The bounds of the Marcum *Q*-function

We compare Sun *et al.* (2010) proposed bounds for the normalized Nuttall *Q*-Function of the order $\mu, \nu \geq 0$ with existing bounds and with Dias and Nunes (2014) algorithm for the selected parameters.

Fig. 6.1 shows the bounds of $Q_\nu(a,b)$ with different values of a, i.e., $a \in \{1, 2.5, 4\}$, when $\nu = 2$. Fig. 6.2 plots the values of $Q_\nu(a,b)$ for different values of ν, i.e., $\nu \in \{2, 5, 8\}$, when $a = 2$. Fig. 6.3 addresses the numerical results for the proposed bounds for ν with non-integer order, i.e., $\nu \in \{1.8, 5.1\}$, when $a = 1.8$.

6.2.2 The bounds of the Nuttall *Q*-function

For the bounds of the normalized Nuttall *Q*-function, Fig. 6.4 shows the bounds of $\mathcal{Q}_{\mu,\nu}(a,b)$ and its bounds versus b for different values of a, i.e., $a = 1, 3$, when $\mu = 4$ and $\nu = 2$. Fig. 6.5 shows the bounds of $\mathcal{Q}_{\mu,\nu}(a,b)$ with non-integer order, where $\mu - \nu = 2$, $\nu = 1.7, 5.2$ and $a = 1$.



(a) (b)

Figure 6.2: Numerical results for $Q_\nu(a, b)$ and the proposed bounds versus b for $\nu \in \{2, 5, 8\}$ and $a = 2$. (a) The bounds $Q_{\nu-LB1}(a, b)$, $Q_\nu^{DN2014}(a, b)$, $Q_{\nu-UB1}(a, b)$ and $Q_{\nu-UB2}(a, b)$. The bounds $Q_{\nu-LB2}(a, b)$, $Q_\nu^{DN2014}(a, b)$, $Q_{\nu-UB3}(a, b)$ and $Q_{\nu-UB4}(a, b)$

6.3 Option pricing model results

6.3.1 CEV results

After extensive computational experiments, Larguinho *et al.* (2013) concluded that the Gamma series method is an appropriate choice for the benchmark of the non-central chi-square distribution function computation. Furthermore, the authors conduct thorough testing on alternative methods to compute the non-central chi-square function at the statistic level and option pricing and hedging under the CEV model. Both distribution approaches point Benton and Krishnamoorthy (2003) as offering the best speed-accuracy trade-off for pricing and hedging options under the CEV model. Our study will compare the efficiency of the aforementioned Benton and Krishnamoorthy (2003) (BK03) method against the recent Gil *et al.* (2014) (G14), while having the Gamma series method serving as benchmark, to understand how quick and accurate those competing methods are for the purpose of pricing and hedging under the CEV method[3]. We concentrate our analysis on call options although the same reasoning could be extended to put option contracts. We rely on the 2500 randomly generated option contract parameters described in Larguinho *et al.* (2013), and further considerations therein, leaving us with 2474 option contracts to test[4]. Since CPU time for a

[3] We provide the Fortran code in Appendix B.

[4] Option contracts with $2x \geq 5000$ and $2y \geq 5000$ were excluded since it is well known – see for instance Schroder (1989) – that speed, overflow and underflow problems could arise when $2x$ and $2y$ are very large.

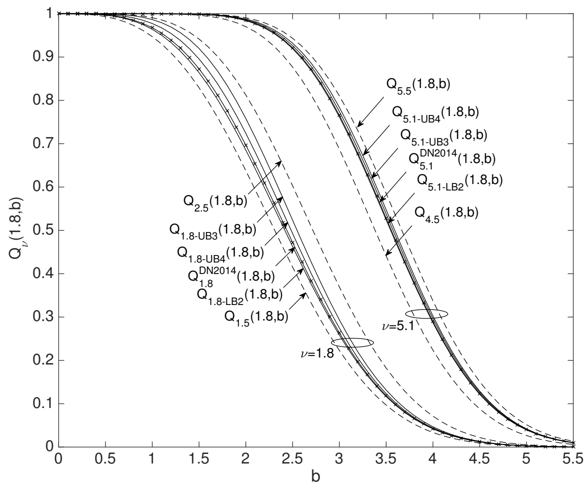


Figure 6.3: Numerical results for the bounds of $Q_\nu(a, b)$ with non-integer order, where $\nu = 1.8, 5.1$ and $a = 1.8$ Crosses: Dias and Nunes (2014) algorithm. Dashed line: previous bounds. Solid line: some of Sun *et al.* (2010) proposed bounds including $Q_{\nu-LB2}(a, b)$, $Q_\nu^{DN2014}(a, b)$, $Q_{\nu-UB3}(a, b)$ and $Q_{\nu-UB4}(a, b)$.

single run on the 2474 contract set is very small, we have performed the analysis 1000 times for the whole set of contracts.

For the Gamma series method, we have considered the stopping approach described in Knüsel and Bablok (1996) and Knüsel (1986). Caution has been taken to perform the series summation in the backward and forward direction for $P_\mu(x, y)$ and $Q_\mu(x, y)$ to avoid numerical cancellation errors.

We have tested the maximum error in the computation of the incomplete Gamma function ratios $P(a, x) = \gamma(a, x)/\Gamma(a)$ and $Q(a, x) = \Gamma(a, x)/\Gamma(a)$ while using Gil *et al.* (2012) **IncgamFI**, subroutine **incgam**, to compute the cumulative distribution function of χ_v^2 according to series expansions in equations (5.4) and (5.7) in chapter 5. Using the relations in Gil *et al.* (2012, Eqs. 2.8 to 2.10), implemented in subroutine **checkincgam**, we have a maximum error of 1.44E-13 and an error of less than 1E-16, 1E-15 and 1E-14 for 56.4%, 93.6% and 99.9% of the cumulative distribution functions respectively, for the whole set of 2474 contracts (4948 non-central chi-square distribution functions).

We have written a Fortran 90 module called **GammaKnueselBablok** that defines a *primary function* as the smallest of $P_\mu(x, y)$ and $Q_\mu(x, y)$ to be computed first and a *secondary function* to be computed based on the relation $P_\mu(x, y) + Q_\mu(x, y) = 1$. Following Gil *et al.* (2014), the transition in the (x, y) quarter plane from small values of $Q_\mu(x, y)$ to values close to unity occurs for large values of μ, x, y across the line $y = x + \mu$, and above this line in the (x, y) quarter plane, $Q_\mu(x, y)$ is taken as the primary function. Below this line, the complementary function $P_\mu(x, y)$ is taken as the primary function, thus avoiding serious

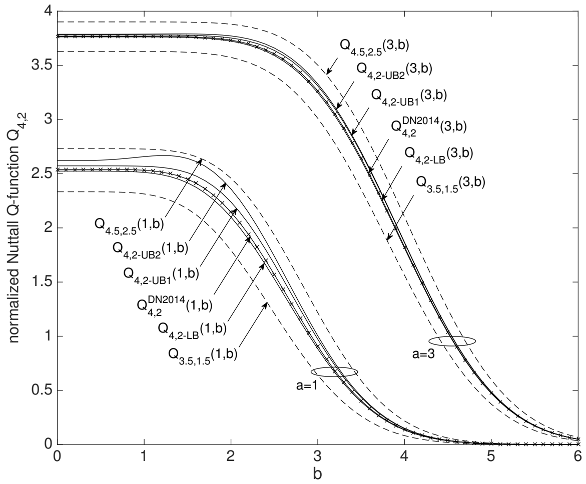


Figure 6.4: Numerical results for the normalized Nuttall Q-function $\mathcal{Q}_{\mu,\nu}(a,b)$ and its bounds versus b for different values of $a = 1, 3$, when $\mu = 4$ and $\nu = 2$. Crosses: Dias and Nunes (2014) algorithm. Dashed line: previous bounds. Solid line: Sun *et al.* (2010) new bounds including $\mathcal{Q}_{\mu,\nu-LB}(a,b)$, $\mathcal{Q}_{\mu,\nu-UB1}(a,b)$ and $\mathcal{Q}_{\mu,\nu-UB2}(a,b)$

cancellation problems that can arise when $P_\mu(x,y)$ is simply computed as $1 - Q_\mu(x,y)$ when $Q_\mu(x,y)$ is close to 1.

For the BK03 iterative approach, we have used a maximum of 10,000 iterations in the convergence procedure, while adhering to a 1E-15 demanded accuracy.

For the G14 approach, we have used the **GammaCHI** module as provided by Gil *et al.* (2015).

Table 6.2 shows values for the differences in call option prices under the CEV assumption using the iterative procedure of Benton and Krishnamoorthy (2003) (BK03) and Gil *et al.* (2014) (G14) compared against the benchmark based on the Gamma series approach, which took 758.00 seconds to compute 1000 times the whole set of 2474 call option prices. MaxAE, MaxRE, RMSE, MeanAE, and k2 denote, respectively, the maximum absolute error, the maximum relative error, the root mean absolute error, the mean absolute error, and the number of times the absolute difference between the two methods exceeds \$0.01.

The results in table 6.2 show that G14 is roughly 25% faster than BK03 method while being more accurate. Both methods return $k_2 = 0$.

We can conclude that G14 offers a best speed-accuracy trade-off over the BK03 method, poising itself as a relevant finding for future work aiming to value option contracts under the one-dimensional CEV model.

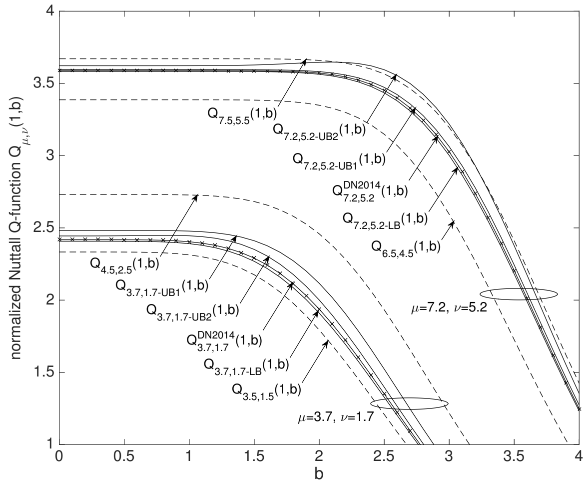


Figure 6.5: Numerical results for the normalized Nuttall Q-function $\mathcal{Q}_{\mu,\nu}(a, b)$ with non-integer order, where $\mu - \nu = 2$, $\mu = 1.7, 5.2$ and $a = 1$. Crosses: Dias and Nunes (2014) algorithm. Dashed line: previous bounds. Solid line: Sun *et al.* (2010) new bounds including $\mathcal{Q}_{\mu,\nu-LB}(a, b)$, $\mathcal{Q}_{\mu,\nu-UB1}(a, b)$ and $\mathcal{Q}_{\mu,\nu-UB2}(a, b)$

Table 6.2: Differences in call option prices using each alternative method for computing the non-central chi-square distribution compared against a benchmark based on the Gamma series approach.

Method	MaxAE	MaxRE	RMSE	MeanAE	CPU time	k_2
BK03	3.21E-10	1.08E-09	1.15E-11	1.68E-12	12.37	0
G14	8.82E-11	3.74E-13	2.66E-12	5.90E-13	9.11	0

Summary of the differences in call option prices under the CEV model using the iterative procedures of Benton and Krishnamoorthy (2003) and Gil *et al.* (2014) compared against a benchmark based on the Gamma series approach, which took a CPU time of 758.00 seconds to compute 1000 times the whole set of 2474 call option prices. The second rightmost column of the table reports the CPU time for computing 1000 times the 2474 call option prices under each alternative method. MaxAE, MaxRE, RMSE, MeanAE, and k2 denote, respectively, the maximum absolute error, the maximum relative error, the root mean absolute error, the mean absolute error, and the number of times the absolute difference between the two methods exceeds \$0.01.

6.3.2 JDCEV results

In this section we perform tests of Dias and Nunes (2014) and Gil *et al.* (2013) Nuttall function iterative algorithms for the computation of option prices under Carr and Linetsky (2006) framework. We choose to compute put over call option prices to include the recovery part of the European-style contract defined in equations (4.16) and (4.17), and the respective raw moment. Since Gil *et al.* (2013) only offers the computation of $\widetilde{Q}_{p,v}(\lambda, x)$ through Gil *et al.* (2013, Eq. 9), and as we rely on the relations defined in equations (5.25) and (5.26) to compute the truncated moments defined in equations (4.10) and (4.11), we have computed European-style call option prices and we have used the put call-parity

$c_{t_0}(S,K,T) - p_{t_0}(S,K,T) = e^{\int_t^T q(u)du} S - e^{\int_t^T r(u)du} K$, that arises from the identity defined in equation (5.26), to compute its counterpart put contract.

For the definition of the data-set to be used in the tests, we have followed the method described in Broadie and Detemple (1996). For simplicity, we have tested 2,500 options contracts following a time-homogeneous model with randomly distributed constant parameters. We have fixed the spot price $S = 100$ and, with uniform probability within each interval, the strike price $K \in [70, 130]$, time to maturity $T \in [0.1, 1.0]$ with probability 0.75 and $T \in]1.0, 5.0]$ with probability 0.25. For the instantaneous volatility parameter defined in Carr and Linetsky (2006, Eq. 4.1), we have made the local volatility vary with uniform probability $\sigma \in [0.1, 0.6]$ and fixed $\beta = -1$. For the default intensity function parameters defined in Carr and Linetsky (2006, Eq. 4.2), we set the sensitivity of the default arrival rate to vary uniformly $c \in [0, 1]$ and $b \in [0, 0.02]$. These parameters have been chosen to satisfy the condition $p + v >= 0$, remarked in Carr and Linetsky (2006, Lemma 5.1).

The choice of a benchmark relies upon the computation of the Gamma series defined in equations (5.25) and (5.26) truncated to ensure a $1E - 15$ accuracy using a stopping rule similar to the one proposed by Knüsel and Bablok (1996), that we have implemented in a Fortran 90 routine **thetaGammaSeries**. The incomplete Gamma functions used in the series summation are the ones provided in Gil *et al.* (2012) module **IncgamFI**.

For the G13 approach, we have used the **NuttallF** module[5] for the computation of the Nuttal Q-function and respective truncated moments, in terms of incomplete Gamma function ratios as described in Gil *et al.* (2013, Eq. 9).

Different approaches have been used to compute the raw moments of equation (4.17). For the benchmark based in the Gamma series routine **thetaGammaSeries**, we defined the raw moment as the sum of two truncated moments as defined in equation (5.28). For the approaches of G13 and DN14, we have relied on $M(p; v, \lambda) = \lim_{x \downarrow 0} \Phi_{+1}(p, x; v, \lambda)$ presented in Dias and Nunes (2016, Eq. 40) and so we have computed $M(p; v, \lambda)$ as $\Phi_{+1}(p, \mathit{realmin}; v, \lambda)$, where *realmin* is the *machine epsilon*[6].

Table (6.3) shows values for the differences in put option prices under the JDCEV assumption using the iterative procedure of Gil *et al.* (2013) (G13) and Dias and Nunes (2014) (DN14) compared against the benchmark based on the Gamma series approach, which took 600.00 seconds to compute 1000 times the whole set of 2500 put option prices. MaxAE, MaxRE, RMSE, MeanAE, and k2 denote, respectively, the maximum absolute error, the

[5] Kindly provided by the authors and including revised versions for efficiency of the **IncgamFI** module for the computation of the incomplete Gamma function ratios used in Gil *et al.* (2013, Eq. 9).

[6] We provide the Fortran code in Appendix C.

Table 6.3: Differences in put option prices using each alternative method for computing the truncated moments of the non-central chi-square distribution compared against a benchmark based on the Gamma series approach.

Method	MaxAE	MaxRE	RMSE	MeanAE	CPU time	k_2
G13	2.95E+01	1.92E+02	2.85E+00	0.45E+00	400.0	166
DN14	8.15E-11	3.41E-10	4.29E-12	1.32E-12	130.0	0

Summary for the computation values of European-style put option prices under the time-homogeneous JDCEV model for the parameter constellation $S = 100$ and, with uniform probability within each interval, $K \in [70, 130]$, $T \in [0.1, 1.0]$ with probability 0.75 and $T \in]1.0, 5.0]$ with probability 0.25, $\sigma \in [0.1, 0.6]$, $\beta = -1$, $c \in [0, 1]$ and $b \in [0, 0.02]$ (the parameters have been chosen to satisfy the condition $p + v >= 0$). The computational results are obtained via the implementation of the explicit solutions (4.11), (4.17) and (4.16) using the iterative procedures of Dias and Nunes (2014) and Gil *et al.* (2013) compared against a benchmark based on the Gamma series approach, which took a CPU time of 600.00 seconds to compute 1000 times the set of 2,500 put option prices. The second rightmost column of the table reports the CPU time for computing 1000 times the 2,500 put option prices under each alternative method. MaxAE, MaxRE, RMSE, MeanAE, and k2 denote, respectively, the maximum absolute error, the maximum relative error, the root mean absolute error, the mean absolute error, and the number of times the absolute difference between the two methods exceeds \$0.01.

maximum relative error, the root mean absolute error, the mean absolute error, and the number of times the absolute difference between the two methods exceeds \$0.01.

The results in table 6.3 show that DN14 is roughly 70% faster than G13 method while being more accurate. While DN14 never exceeds the difference of \$0.01 against the benchmark, G13 returns $k_2 = 166$, representing 7% of contracts exceeding the pre-defined threshold. Inspection of the input parameters reveal that for high values of the non-centrality parameter ($\lambda > 80$), G13 results significantly diverge from the benchmark.

We can conclude that DN14 offers a best speed-accuracy trade off over the G13 method. Furthermore, as we have defined input parameters for the JDCEV framework believed to be consistent with those used in the practice, and considering the differences observed in G13 against the benchmark, we find G13 method less suitable for the computation of option prices under this framework. On the other hand, DN14 poises itself as an efficient and accurate approach to use under the studied option pricing framework.

Chapter 7

JDCEV Hedge Ratios

In this section we offer new analytical solutions for the JDCEV delta hedge ratios for the recovery parts of the put – offered in Ruas *et al.* (2013, Eqs. 38 and 40) using Kummer confluent hypergeometric functions of the first kind – resorting to the series solutions in equation (5.37) for the derivation of p-th moment about zero of a random variable $X \sim \chi^2(v, \lambda)$.

7.1 Delta of the recovery part of the put (4.17), under the fractional recovery of treasury assumption

$$\begin{aligned}\frac{\partial SP\,(S,K,T)}{\partial S} &= -\frac{K}{S_{t_0}} e^{-(r+b)(T-t_0)} \left(\frac{x^2(S_{t_0})}{\tau}\right)^{\frac{1}{2|\beta|}} \\ &\times \left[M\left(-\frac{1}{2\,|\beta|}, \frac{k^2(K)}{\tau}; \delta_+, \frac{x^2(S_{t_0})}{\tau}\right) \left(1 - |\beta|\,\frac{x^2(S_{t_0})}{\tau}\right) \right. \\ &\left. + 2\,|\beta|\,\widetilde{M}\left(-\frac{1}{2\,|\beta|}, \frac{k^2(K)}{\tau}; \delta_+, \frac{x^2(S_{t_0})}{\tau}\right)\right],\end{aligned} \tag{7.1}$$

where M is the p-th moment about zero of a random variable $X \sim \chi^2(v, \lambda)$ as defined in the series solution (5.37) and,

$$\widetilde{M}\,(p; v, \lambda) = 2^p \sum_{i=0}^{\infty} \frac{e^{-\frac{\lambda}{2}}\left(\frac{\lambda}{2}\right)^i}{(i-1)!} \frac{\Gamma(p + \frac{v}{2} + i)}{\Gamma(\frac{v}{2} + i)}. \tag{7.2}$$

7.2 Delta of the recovery part of the put (4.19), under the fractional recovery of face value assumption

$$\begin{aligned}
\frac{\partial p^{D}_{t_0}\left(S_{t_0},K,T\right)}{\partial S} = R\int_{t_0}^{T}\frac{e^{-(r+b)(u-t_0)}}{S}&\left[b\left(\frac{x^2(S_{t_0})}{\tau}\right)^{\frac{1}{2|\beta|}}\left(M\left(-\frac{1}{2\left|\beta\right|};\delta_+,\frac{x^2(S_{t_0})}{\tau}\right)\right.\right.\\
&\times\left(1-\left|\beta\right|\frac{x^2(S_{t_0})}{\tau}\right)+2\left|\beta\right|\widetilde{M}\left(-\frac{1}{2\left|\beta\right|};\delta_+,\frac{x^2(S_{t_0})}{\tau}\right)\Bigg)\\
&+ca^2S^{2|\beta|}e^{-2|\beta|(r-q+b)(u-t_0)}\left(\frac{x^2(S_{t_0})}{\tau}\right)^{\frac{1}{2|\beta|}+1}\left(M\left(-\frac{1}{2\left|\beta\right|}-1;\delta_+,\frac{x^2(S_{t_0})}{\tau}\right)\right.\\
&\left.\left.\left(1+4\left|\beta\right|-\left|\beta\right|\frac{x^2(S_{t_0})}{\tau}\right)+2\left|\beta\right|\widetilde{M}\left(-\frac{1}{2\left|\beta\right|}-1;\delta_+,\frac{x^2(S_{t_0})}{\tau}\right)\right)\right]du,
\end{aligned} \tag{7.3}$$

where M is the p-th moment about zero of a random variable $X\sim\chi^2(v,\lambda)$ as defined in the series solution (5.37) and $\widetilde{M}$ is as defined in equation (7.2).

Chapter 8

Conclusions

With this work we have tested the recent offerings in the literature to compute the non-central chi-square distribution and its related functions under the CEV option pricing model of Cox (1975) and the JDCEV framework of Carr and Linetsky (2006). We gave particular emphasis to the work of Sun *et al.* (2010), Gil *et al.* (2012), Gil *et al.* (2013), Dias and Nunes (2014), Gil *et al.* (2014) and Gil *et al.* (2015).

We started by testing Gil *et al.* (2012) (G12) Fortran 90 **IncgamFI** incomplete Gamma function ratios module. As we relied on Gil *et al.* (2012) package interchangeably with Matlab **gammainc** function for the computation of these rations, we found it relevant to test it against each other. The results showed that both methods agree on a double precision of 1E-15 accordance, required in the subsequent tests.

We have computed call option prices under the CEV framework for 2,474 contracts, using the iterative procedure of Benton and Krishnamoorthy (2003) (BK03) and Gil *et al.* (2014) (G14) compared against the benchmark based on the Gamma series approach to compute non-central chi-square distribution function. The results show that G14 is roughly 25% faster than BK03 method while being more accurate. Both methods return no significant differences against a pre-defined threshold of \$0.01. We conclude that G14 offers a best speed-accuracy trade off over the BK03 method, poising itself as a relevant finding for future work aiming to value option contracts under the one-dimensional CEV model.

We have computed put option prices under the JDCEV framework for 2,500 contracts, using the iterative procedure of Gil *et al.* (2013) (G13) and Dias and Nunes (2014) (DN14), compared against a benchmark based on a Gamma series approach. The results show that DN14 is roughly 70% faster than G13 while being more accurate. While DN14 never exceeds the difference of \$0.01 against the benchmark, G13 returns 7% of contracts exceeding that pre-defined threshold. Inspection of the input parameters reveal that for high values of the non-centrality parameter ($\lambda > 80$), G13 returns widely inaccurate results. We can conclude that DN14 offers a best speed-accuracy trade off over the G13 method. Furthermore, as we have defined input parameters for the JDCEV framework believed to be consistent with those used in the practice, and considering the the differences observed in G13 against the benchmark, we find G13 method less suitable for the computation of option prices under this framework. On the other hand, DN14 poises itself as a very efficient and accurate approach to use under the studied option pricing framework.

Additionally, we have tested Dias and Nunes (2014) algorithm against Sun *et al.* (2010) proposed tight bounds for the computation of the marcum and Nuttall Q-function. Overall, we conclude that Dias and Nunes (2014) results lie exactly inside these newly proposed tight bounds, corroborating the robustness of Dias and Nunes (2014) algorithm.

Lastly, we offered new analytical solutions for the JDCEV delta hedge ratios for the recovery parts of the put resorting to a series solutions for the derivation of p-th moment about zero of a random variable $X \sim \chi^2(v, \lambda)$.

Bibliography

Abramowitz, M. & Stegun, I. 1972. ***Handbook of mathematical functions***. New York: Dover.

Annamalai, A. & Tellambura, C. 2001. Cauchy-Schwarz bound on the generalized Marcum Q-function with applications. ***Wireless Communications and Mobile Computing***, 1(2): 243–253.

Ball, C. A. & Torous, W. N. 1983. Bond price dynamics and options. ***Journal of Financial & Quantitative Analysis***, 18(4): 517–531.

Baricz, Á. & Sun, Y. 2009. New bounds for the generalized marcum Q-Function. ***IEEE Transactions on Information Theory***, 55(7): 3091–3100.

Bekaert, G. & Wu, G. 2000. Asymmetric volatility and risk in equity markets. ***Review of Financial Studies***, 13(1): 1–42.

Benton, D. & Krishnamoorthy, K. 2003. Computing discrete mixtures of continuous distributions: Noncentral chisquare, noncentral t and the distribution of the square of the sample multiple correlation coefficient. ***Computational Statistics & Data Analysis***, 43(2): 249–267.

Black, F. 1975. Fact and fantasy in the use of options. ***Financial Analysts Journal***, 31(4): 36–72.

Black, F. 1976. Studies of stock price volatility changes. ***Proceedings of the 1976 Meetings of the American Statistical Association***: 177–181. Washington, D.C.: American Statistical Association.

Black, F. & Cox, J. C. 1976. Valuing corporate securities: Some effects of bond indenture provisions. ***Journal of Finance***, 31(2): 351–367.

Black, F., Derman, E. & Toy, W. 1990. A one-factor model of interest rates and its application to treasury bond options. ***Financial Analysts Journal***, 46(1): 33.

Black, F. & Karasinski, P. 1991. Bond and option pricing when short rates are lognormal. ***Financial Analysts Journal***, 47(4): 52–59.

Black, F. & Scholes, M. S. 1973. The pricing of options and corporate liabilities. ***Journal of Political Economy***, 81(3): 637–654.

Brace, A., Gatarek, D. & Musiela, M. 1997. The market model of interest rate dynamics. ***Mathematical Finance***, 7(2): 127–155.

Breiman, L. 1986. ***Probability and stochastic processes*** (2nd ed.). CA: Scientific Press.

Brennan, M. J. & Schwartz, E. S. 1977. Savings bonds, retractable bonds and callable bonds. ***Journal of Financial Economics***, 5(1): 67–88.

Brennan, M. J. & Schwartz, E. S. 1979. A continuous time approach to the pricing of bonds. ***Journal of Banking & Finance***, 3(2): 133–155.

Brennan, M. J. & Schwartz, E. S. 1980. Analyzing convertible bonds. ***Journal of Financial and Quantitative Analysis***, 15(4): 907–929.

Brennan, M. J. & Schwartz, E. S. 1982. An equilibrium model of bond pricing and a test of market efficiency. ***Journal of Financial and Quantitative Analysis***, 17(03): 301–329.

Brigo, D. & Alfonsi, A. 2005. Credit default swap calibration and derivatives pricing with the SSRD stochastic intensity model. ***Finance & Stochastics***, 9(1): 29–42.

Brigo, D. & Mercurio, F. 2006. ***Interest rate models - Theory and practice: With smile, inflation and credit*** (2nd ed.). Berlin: Springer Finance.

Broadie, M. & Detemple, J. 1996. American option valuation: new bounds, approximations, and a comparison of existing methods. ***Review of Financial Studies***, 9(4): 1211–1250.

Campbell, J. Y. & Taskler, G. B. 2003. Equity volatility and corporate bond yields. ***Journal of Finance***, 58(6): 2321–2350.

Carr, P. & Linetsky, V. 2006. A jump to default extended CEV model: an application of Bessel processes. ***Finance and Stochastics***, 10(3): 303–330.

Chan, K. C., Karolyi, G., Longstaff, F. & Sanders, A. 1992. An empirical comparison of alternative models of the short-term interest rate. ***The Journal of Finance***, 47(3): 1209–1227.

Chen, R.-R. & Lee, C.-F. 2010. A constant elasticity of variance (CEV) family of stock price distributions in option pricing, review, and integration. In Lee, C.-F., Lee, A. C. & Lee, J. (Eds.). ***Handbook of Quantitative Finance and Risk Management***, 1615–1625. Boston, MA: Springer US.

Consigli, G. 2004. ***Credit default swaps and equity volatility: Theoretical modeling and market evidence.*** Workshop on portfolio optimisation and option pricing, Dept. of Mathematics University Ca'Foscari of Venice.

Courtadon, G. 1982. The pricing of options on default free bonds. ***Journal of Financial & Quantitative Analysis***, 17(1): 75–100.

Cox, J. C. 1975. ***Notes on option pricing I: Constant elasticity of diffusions.*** Unpublished draft, Stanford University.

Cox, J. C., Ingersoll Jr., J. E. & Ross, S. A. 1980. An analysis of variable rate loan contracts. ***Journal of Finance***, 35(2): 389–403.

Cox, J. C., Ingersoll, J. E. & Ross, S. A. 1981. A re-examination of traditional hypotheses about the term structure of interest rates. ***Journal of Finance***, 36(4): 769–799.

Cox, J. C., Ingersoll, J. E. & Ross, S. A. 1985. A theory of the term structure of interest rates. ***Econometrica***, 53(2): 385–408.

Cox, J. C., Ingersoll, J. E. & Ross, S. A. 1985. An intertemporal general equilibrium model of asset prices. ***Econometrica***, 53(2): 363–384.

Cox, J. C. & Ross, S. A. 1976. The valuation of options for alternative stochastic processes. ***Journal of Financial Economics***, 3(1–2): 145–166.

Cremers, M., Driessen, J., Maenhout, P. & Weinbaum, D. 2008. Individual stock-option prices and credit spreads. ***Journal of Banking & Finance***, 32(12): 2706–2715.

Culbertson, J. M. 1957. The term structure of interest rates. ***The Quarterly Journal of Economics***, 71(4): 485–517.

Davydov, D. & Linetsky, V. 2001. Pricing and hedging path-dependent options under the CEV process. ***Management Science***, 47(7): 949–965.

Dennis, P. & Mayhew, S. 2002. Risk-neutral skewness: Evidence from stock options. ***Journal of Financial and Quantitative Analysis***, 37(03): 471–493.

Dias, J. C. & Nunes, J. P. 2011. Pricing real options under the CEV diffusion. ***Journal of Futures Markets***, 31(3): 230–250.

Dias, J. C. & Nunes, J. P. 2014. A fast and accurate algorithm for computing truncated moments of a noncentral χ^2 random variable with applications to option pricing: 1–19.

Dias, J. C. & Nunes, J. P. 2016. ***Universal algorithm for computing Nuttall and generalized Marcum Q-functions and moments of a noncentral $\chi 2$ random variable.*** Working Paper, BRU-UNIDE and ISCTE-IUL Business School.

DiDonato, A. R. & Morris Jr., A. H. 1986. Computation of the incomplete Gamma function ratios and their inverse. ***ACM Transactions on Mathematical Software***, 12(4): 377–393.

Ding, C. G. 1992. Computing the non-central χ^2 distribution function. ***Journal of the Royal Statistical Society: Series C (Applied Statistics)***, 41(2): 478–482.

Dothan, L. 1978. On the term structure of interest rates. ***Journal of Financial Economics***, 6(1): 59–69.

Duffie, D. & Kan, R. 1996. A yield-factor model of interest rates. ***Mathematical Finance***, 6(4): 379–406.

Duffie, D. & Singleton, K. J. 1999. Modeling term structures of defaultable bonds. ***Review of Financial Studies***, 12(4).

Dyrting, S. 2004. Evaluating the Noncentral Chi-Square Distribution for the Cox-Ingersoll-Ross Process. ***Computational Economics***, 24(1): 35–50.

Elliott, J. W. & Baier, J. R. 1979. Econometric models and current interest rates: How well do they predict future rates? ***Journal of Finance***, 34(4): 975–986.

Elliott, R. J., Jeanblanc, M. & Yor, M. 2000. On models of default risk. ***Mathematical Finance***, 10(2): 179–195.

Emanuel, D. C. & MacBeth, J. D. 1982. Further results on the constant elasticity of variance call option pricing model. ***Journal of Financial & Quantitative Analysis***, 17(4): 533–554.

Fabozzi, F. J. & Mann, S. V. 2005. ***The handbook of fixed income securities*** (7th ed.). New York: McGraw-Hill.

Fama, E. F. 1984. Term premiums in bond returns. ***Journal of Financial Economics***, 13(4): 529–546.

Fama, E. F. 1984. The information in the term structure. ***Journal of Financial Economics***, 13(4): 509–528.

Farebrother, R. W. 1987. Algorithm AS 231. ***Journal of the Royal Statistical Society: Series C (Applied Statistics)***, 36(3): 402.

Feller, W. 1951. Two singular diffusion problems. ***Annals of Mathematics***, 54(1): 173–182.

Fisher, I. 1896. Appreciation and interest. ***Publications of the American Economic Association***, 11(4): 331–442.

Fisher, R. A. 1928. The general sampling distribution of the multiple correlation coefficient. ***Proceedings of the Royal Society of London: Series A, Containing Papers of a Mathematical and Physical Character***, 121(788): 654–673.

Fong, H. G. & Vasicek, O. A. 1991. Fixed-income volatility management. ***Journal of Portfolio Management***, 17(4): 41–46.

Fraser, D. A. S., Wu, J. & Wong, A. C. M. 1998. An approximation for the noncentral chi-squared distribution. ***Communications in Statistics - Simulation and Computation***, 27(2): 275–287.

Gautschi, W. 1979. A computational procedure for incomplete gamma functions. ***ACM Transactions on Mathematical Software***, 5(4): 466–481.

Geman, H. & Shih, Y. F. 2009. Modeling commodity prices under the CEV model. ***Journal of Alternative Investments***, 11(3): 65–84.

Gil, A., Segura, J. & Temme, N. M. 2012. Efficient and accurate algorithms for the computation and inversion of the incomplete gamma function ratios. ***SIAM Journal on Scientific Computing***, 34(6): 2965–2981.

Gil, A., Segura, J. & Temme, N. M. 2013. On the computation of moments of the partial non-central chi-square distribution function. ***Proceedings of the International Conference "Applications of Mathematics 2013".*** 2(2): 98–103.

Gil, A., Segura, J. & Temme, N. M. 2014. Algorithm 939: Computation of the Marcum Q-function. ***ACM Transactions on Mathematical Software***, 40(3).

Gil, A., Segura, J. & Temme, N. M. 2015. GammaCHI: A package for the inversion and computation of the gamma and chi-square cumulative distribution functions (central and noncentral). ***Computer Physics Communications***, 191(1): 132–139.

Glasserman, P. & Merener, N. 2001. Cap and swaption approximations in Libor market models with jumps. ***Journal of Computational Finance***, 7(1): 1–11.

Glasserman, P. & Merener, N. 2003. Numerical solution of jump-diffusion LIBOR market models. ***Finance and Stochastics***, 7(1): 1–27.

Heath, D., Jarrow, R. A. & Morton, A. 1992. Bond Pricing and the Term Structure of Interest Rates: A New Methodology for Contingent Claims Valuation. ***Econometrica (1986-1998)***, 60(1): 77.

Heston, S. L. 1993. A closed-form solution for options with stochastic volatility with applications to bond and currency options. ***Review of Financial Studies***, 6(2): 327–343.

Hicks, J. R. 1939. ***Value and capital: An inquiry into some fundamental principles of economic theory***. Oxford: Clarendon Press.

Ho, T. S. Y. & Lee, S.-B. 1986. Term structure movements and pricing interests rate contingent claims. ***Journal of Finance***, 41(5): 1011–1029.

Hsu, Y., Lin, T. & Lee, C. 2008. Constant elasticity of variance (CEV) option pricing model: Integration and detailed derivation. ***Mathematics and Computers in Simulation***, 79(1): 60–71.

Hull, J. & White, A. 1990. Pricing interest-rate derivative securities. ***Review of Financial Studies***, 3(4): 573–592.

Ingersoll Jr., J. E. & Ross, S. A. 1992. Waiting to invest: Investment and uncertainty. ***Journal of Business***, 65(1): 1.

Jackwerth, J. C. & Rubinstein, M. 2012. Recovering stochastic processes from option prices. In Batten, J. A. & Wagner, N. (Eds.). ***Derivative securities pricing and modelling: Contemporary studies in economic and financial analysis***, vol. 94: 123–153. Bingley, UK: Emerald Group Publishing Limited.

Jamshidian, F. 1989. An exact bond option formula. ***Journal of Finance***, 44(1): 205–209.

Jamshidian, F. 1997. LIBOR and swap market models and measures. ***Finance and Stochastics***, 1(4): 293–330.

Jarrow, R. A. 2003. A robust test of Merton's structural model for credit risk. ***Journal of Risk***, 6(1): 39–58.

Jarrow, R. A., Lando, D. & Turnbull, S. M. 1997. A Markov model for the term structure of credit risk spreads. ***Review of Financial Studies***, 10(2): 481–523.

Jarrow, R. A. & Protter, P. 2004. Structural versus reduced form models: A new information based perpspective. ***Investment Management***, 2(2): 1–10.

Jarrow, R. A. & Turnbull, S. M. 1995. Pricing Derivatives on Financial Securities Subject to Credit Risk. ***Journal of Finance***, 50(1): 53–85.

Johnson, N. L., Kotz, S. & Balakrishnan, N. 1995. ***Continuous univariate distributions*** (2nd ed.), vol. 2. New York: John Wiley & Sons.

Kapinas, V. M., Mihos, S. K. & Karagiannidis, G. K. 2009. On the Monotonicity of the Generalized Marcum and Nuttall Q-Functions. ***IEEE Transactions on Information Theory***, 55(8): 3701–3710.

Keynes, J. M. 1930. ***A treatise on money***, vol. 2. London: Macmillan.

Keynes, J. M. 1936. ***The general theory of employment, interest and money***. London: Macmillan.

Knüsel, L. & Bablok, B. 1996. Computation of the noncentral gamma distribution. ***SIAM Journal on Scientific Computing***, 17(5): 1224–1231.

Knüsel, L. 1986. Computation of the chi square and Poisson distribution. ***SIAM Journal on Scientific Statistical Computing***, 7(3): 1022–1036.

Lando, D. 2009. ***Credit risk modeling: Theory and applications***. Princeton University Press.

Larguinho, M., Dias, J. C. & Braumann, C. A. 2013. On the computation of option prices and Greeks under the CEV model. ***Quantitative Finance***, 13(6): 907–917.

Li, R. & Kam, P. Y. 2006. ***Computing and bounding the generalized Marcum Q-Function via a geometric approach.*** 2006 IEEE International Symposium on Information Theory.

Longstaff, F. 1989. A nonlinear general equilibrium model of the term structure of interest rates. ***Journal of Financial Economics***, 23(2): 195–224.

Longstaff, F. 1993. The valuation of options on coupon bonds. ***Journal of Banking & Finance***, 17(1): 27–42.

Longstaff, F. & Schwartz, E. S. 1992. Interest rate volatility and the term structure: A two-factor general equilibrium model. ***The Journal of Finance***, 47(4): 1259–1282.

Longstaff, F. A. & Schwartz, E. S. 1995. A simple approach to valuing risky Ffxed and floating rate debt. ***Journal of Finance***, 50(3): 789–819.

Lutz, F. A. 1940. The structure of interest rates. ***Quarterly Journal of Economics***, 55(1): 36–63.

Macaulay, F. R. 1938. ***Some theoretical problems suggested by the movements of interest rates, bond yields and stock prices in the United States since 1856***. New York: National Bureau of Economic Research, Inc.

MacBeth, J. D. & Merville, L. J. 1979. An empirical examination of the Black-Scholes call option pricing model. ***Journal of Finance***, 34(5): 1173–1186.

MacBeth, J. D. & Merville, L. J. 1980. Tests of the Black-Scholes and Cox call option valuation models. ***The Journal of Finance***, 35(2): 285–301.

Madan, D. & Unal, H. 1998. Pricing the risks of default. ***Review of Derivatives Research***, 2(2-3): 121–160.

Malkiel, B. G. 1966. ***The term structure of interest rates: expectations and behavior patterns***. Princeton: Princeton University Press.

Marchand, E. 1996. Computing the moments of a truncarted noncentral chi-square distribution. ***Journal of Statistical Computation and Simulation***, 55(1-2): 23–29.

Marcum, J. I. 1960. +A statistical theory of target detection by pulsed radar.+. ***IRE Transactions on Information Theory***, 6(2): 59–267.

Mercurio, F. & Moraleda, J. M. 2000. An analytically tractable interest rate model with humped volatility. ***European Journal of Operational Research***, 120(1): 205–214.

Merton, R. C. 1973. Theory of rational option pricing. ***Bell Journal of Economics and Management Science***, 4(1): 141–183.

Merton, R. C. 1974. On the pricing of corporate debt: The risk structure of interest rates. ***Journal of Finance***, 29(2): 449–470.

Merton, R. C. 1976. Option pricing when underlying stock returns are discontinuous. ***Journal of Financial Economics***, 3(1–2): 125–144.

Miltersen, K. R., Sandmann, K. & Sondermann, D. 1997. Closed form solutions for term structure derivatives with log-normal interest rates. ***Journal of Finance***, 52(1): 409–430.

Modigliani, F. & Shiller, R. J. 1973. Inflation, rational expectations and the term structure of interest rates. ***Economica***, 40(157): 12–43.

Modigliani, F. & Sutch, R. 1966. Innovations in Interest Rate Policy. ***The American Economic Review***, 56(1/2): 178–197.

Nelson, C. R. 1972. Estimation of term premiums from average yield differentials in the term structure of interest rates. ***Econometrica***, 40(2): 277–287.

Nuttall, A. H. 1972. ***Some integrals involving the Q-Function***: Tech report: Naval Underwater Systems Center. New London, Connecticut.

Penev, S. & Raykov, T. 2000. A Wiener Germ approximation of the noncentral chi square distribution and of its quantiles. ***Computational Statistics***, 15(2): 219–228.

Posten, H. O. 1989. An effective algorithm for the noncentral chi-squared distribution function. ***The American Statistician***, 43(4): 261–263.

Rendleman, R. J. & Bartter, B. J. 1980. The pricing of options on debt securities. ***Journal of Financial and Quantitative Analysis***, 15(01): 11–24.

Richard, S. F. 1978. An arbitrage model of the term structure of interest rates. ***Journal of Financial Economics***, 6(1): 33–57.

Ruas, J. P., Dias, J. C. & Nunes, J. P. V. 2013. Pricing and static hedging of American-style options under the jump to default extended CEV model. ***Journal of Banking & Finance***, 37(11): 4059–4072.

Sankaran, M. 1963. Approximations to the non-central chi-square distribution. ***Biometrika***, 50(1-2): 199–204.

Schaefer, S. & Schwartz, E. S. 1987. Time-dependent variance and the pricing of bond options. ***Journal of Finance***, 42(5): 1113–1128.

Scharf, L. L. & Demeure, C. 1991. ***Statistical signal processing: Detection, estimation, and time series analysis***. Addison-Wesley series in electrical and computer engineering. Addison-Wesley Publishing Company.

Schroder, M. 1989. Computing the constant elasticity of variance option pricing formula. ***The Journal of Finance***, 44(1): 211–219.

Shiller, R. J. 1979. The volatility of long-term interest rates and expectations models of the term structure. ***Journal of Political Economy***, 87(6): 1190.

Shiller, R. J. 1981. Alternative tests of rational expectations models: The case of the term structure. ***Journal of Econometrics***, 16(1): 71–87.

Shiller, R. J. & McCulloch, J. H. 1987. The term structure of interest rates. ***National Bureau of Economic Research Working Paper Series***, No. 2341.

Sun, Y., Baricz, Á. & Zhou, S. 2010. On the monotonicity, log-concavity, and tight bounds of the generalized marcum and Nuttall Q-functions. ***IEEE Transactions on Information Theory***, 56(3): 1166–1186.

Sun, Y. & Zhou, S. 2008. ***Tight bounds of the generalized Marcum Q-Function based on log-concavity.*** Paper presented at IEEE Global Telecommunications Conference, 2008.

Telser, L. G. 1967. A critique of some recent empirical research on the explanation of the term structure of interest rates. ***Journal of Political Economy***, 75(4): 546.

Vasicek, O. 1977. An equilibrium characterization of the term structure. ***Journal of Financial Economics***, 5(2): 177–188.

Vassalou, M. & Xing, Y. 2004. Default risk in equity returns. ***Journal of Finance***, 59(2): 831–868.

Zhang, B. Y., Zhou, H. & Zhu, H. 2009. Explaining credit default swap spreads with the equity volatility and jump risks of individual firms. ***Review of Financial Studies***, 22(12): 5099–5131.

Appendix A

Sun *et al.* (2010) Matlab code

A.1 Sun *et al.* (2010) Marcum *Q*-function

A.1.1 Sun *et al.* (2010) eq. (42) and eq. (38)

```
function [q]=marcum_q(nu,a,b)
    if a==0
    %Sun2010 eq. 42
        q=(gammainc((b^2)/2,nu,'upper')*gamma(nu))/gamma(nu);
    elseif (a>0 && b>=0)
        %Sun2010 eq.38
        sum1=0;
        for k=0:(nu-1.5)
            sum2=0;
            for q=0:k
                sum3=0;
                for i=0:(2*q)
                    sum3=sum3+((1/((a*b)^(2*q-i)*factorial(i)))*((-1)^i*exp((-(b-a)^2)/2)-
                        exp((-(b+a)^2)/2)));
                end
                sum2=sum2+((((-1)^q)*factorial(2*q))/(factorial(k-q)*factorial(q))*sum3);
            end
            sum1=sum1+((b^(2*k))/(2^k)*sum2);
        end
        q=0.5*erfc((b+a)/(sqrt(2)))+0.5*erfc((b-a)/(sqrt(2)))+((1/(a*sqrt(2*pi)))*sum1);
    end
end
```

A.1.2 Sun *et al.* (2010) eq. (52)

```
function [q]=marcum_q_lower1(nu,a,b)
    nu1=floor(nu+0.5)+0.5;
    nu2=floor(nu-0.5)+0.5;
    %Sun2010 eq.52
    q=(marcum_q(nu1,a,b)^(nu-nu2))*(marcum_q(nu2,a,b)^(nu1-nu));
end
```

A.1.3 Sun *et al.* (2010) eq. (61)

```
function [q]=marcum_q_lower2(nu,a,b)
    nu1=floor(nu+0.5)+0.5;
    nu2=floor(nu-0.5)+0.5;
    %Sun2010 eq.61
    q=marcum_q(nu,0,b)+(((marcum_q(nu1,a,b)-marcum_q(nu1,0,b))^(nu-nu2))*((marcum_q(nu2,a,b
        )-marcum_q(nu2,0,b))^(nu1-nu)));
end
```

A.1.4 Sun *et al.* (2010) eq. (55)

```
function [q]=marcum_q_upper1(nu,a,b)
    nu1=floor(nu+0.5)+0.5;
    nu2=floor(nu-0.5)+0.5;
    %Sun2010 eq.55
    q=(marcum_q(nu1,a,b)^(nu1-nu+1))/(marcum_q(nu1+1,a,b)^(nu1-nu));
end
```

A.1.5 Sun *et al.* (2010) eq. (56)

```
function [q]=marcum_q_upper2(nu,a,b)
    nu1=floor(nu+0.5)+0.5;
    nu2=floor(nu-0.5)+0.5;
    %Sun2010 eq. 56
    q=(marcum_q(nu2,a,b)^(nu-nu2+1))/(marcum_q(nu2-1,a,b)^(nu-nu2));
end
```

A.1.6 Sun *et al.* (2010) eq. (62)

```
function [q]=marcum_q_upper3(nu,a,b)
    nu1=floor(nu+0.5)+0.5;
    %Sun2010 eq.62
```

```
    q=marcum_q(nu,0,b)+(((marcum_q(nu1,a,b)-marcum_q(nu1,0,b))^(nu1-nu+1))/((marcum_q(nu1
        +1,a,b)-marcum_q(nu1+1,0,b))^(nu1-nu)));
end
```

A.1.7 Sun *et al.* (2010) eq. (63)

```
function [q] = marcum_q_upper4(nu,a,b)
    nu2=floor(nu-0.5)+0.5;
    %Sun2010 eq.63
    q=marcum_q(nu,0,b)+(((marcum_q(nu2,a,b)-marcum_q(nu2,0,b))^(nu-nu2+1))/((marcum_q(nu2
        -1,a,b)-marcum_q(nu2-1,0,b))^(nu-nu2)));
end
```

A.2 Sun *et al.* (2010) Nuttall *Q*-function

A.2.1 Sun *et al.* (2010) eq. (49)

```
function [q]=std_nuttall_q(mu,nu,a,b)
    m=mu+0.5;
    n=nu+0.5;
    %Sun2010 eq. 49
    sum1=0;
    for k=0:(n-1)
        sum1=sum1+(((pochhammer(n-k,n-1)*(2*a)^k)/(factorial(k)))*term_i(m,n,k,a,b));
    end
    q=((((-1)^n)*((2*a)^(-n+0.5)))/(sqrt(pi)))*sum1;
end
```

A.2.2 Sun *et al.* (2010) eq. (68)

```
function [q]=std_nuttall_q_lower(mu,nu,a,b)
    mu1=floor(mu+0.5)+0.5;
    mu2=floor(mu-0.5)+0.5;
    nu1=floor(nu+0.5)+0.5;
    nu2=floor(nu-0.5)+0.5;
    %Sun2010 eq. 68
    q=(std_nuttall_q(mu1,nu1,a,b)^(nu-nu2))*(std_nuttall_q(mu2,nu2,a,b)^(nu1-nu));
end
```

A.2.3 Sun *et al.* (2010) eq. (69)

```
function [q]=std_nuttall_q_upper1(mu,nu,a,b)
    mu1=floor(mu+0.5)+0.5;
    mu2=floor(mu-0.5)+0.5;
    nu1=floor(nu+0.5)+0.5;
    nu2=floor(nu-0.5)+0.5;
    %Sun2010 eq.69
    q=(std_nuttall_q(mu1,nu1,a,b)^(nu1-nu+1))/(std_nuttall_q(mu1+1,nu1+1,a,b)^(nu1-nu));
end
```

A.2.4 Sun *et al.* (2010) eq. (70)

```
function [q]=std_nuttall_q_upper2(mu,nu,a,b)
    mu1=floor(mu+0.5)+0.5;
    mu2=floor(mu-0.5)+0.5;
    nu1=floor(nu+0.5)+0.5;
    nu2=floor(nu-0.5)+0.5;
    %Sun2010 eq.70
    q=(std_nuttall_q(mu2,nu2,a,b)^(nu-nu2+1))/(std_nuttall_q(mu2-1,nu2-1,a,b)^(nu-nu2));
end
```

A.2.5 Sun *et al.* (2010) eq. (50)

```
function i=term_i(m,n,k,a,b)
    %Sun2010 eq.50
    sum1=0;
    for l=0:(m-n+k)
        sum1=sum1+(binomial(m-n+k,l)*2^((l-1)/2)*a^(m-n+k-l)*((gamma((l+1)/2)+((-1)^(m-n-l-1))*gammainc((b+a)^2/2,(l+1)/2,'upper')*gamma((l+1)/2))-sgn(b-a)^(l+1)*(gamma((l+1)/2)-gammainc((b-a)^2/2,(l+1)/2,'upper')*gamma((l+1)/2))));
    end
    i=((-1)^(k+1))*sum1;
end
```

Appendix B

CEV Fortran code

B.1 Code to test the different algorithms to compute CEV option prices

```
PROGRAM testEuropeanCEV
  USE EuropeanCEV_jcd
  IMPLICIT NONE
  INTEGER, PARAMETER  :: r8 = KIND(0.0d0)
  REAL(r8) :: p, x, n, lambda, start, finish, k2_threshold
  REAL(r8) :: EuropeanCallGamma, EuropeanPutGamma, EuropeanCallBK, EuropeanPutBK, EuropeanCallGST, EuropeanPutGST, &
  EuropeanCallJCD, EuropeanPutJCD, incgam_delta, incgam_delta1, incgam_delta2
  INTEGER :: theta, maxitr, i, j, ierr1, ierr2
  INTEGER, parameter :: iu1=1234
  INTEGER, parameter :: iu2=1235
  INTEGER, parameter :: iu3=1236
  REAL(r8), PARAMETER :: errtol=1.e-15_r8;  ! demanded accuracy
  REAL(r8), dimension(2474)  :: Call_BK, Put_BK, Call_GST, Put_GST, Call_Gamma, Put_Gamma, Call_JCD, Put_JCD, &
   Call_BK_AE, Put_BK_AE, Call_GST_AE, Put_GST_AE, Call_JCD_AE, Put_JCD_AE, &
   Call_BK_k2, Put_BK_k2, Call_GST_k2, Put_GST_k2, Call_JCD_k2, Put_JCD_k2, &
   Call_BK_RE, Put_BK_RE, Call_GST_RE, Put_GST_RE, Call_JCD_RE, Put_JCD_RE,&
   Call_BK_SQE, Put_BK_SQE, Call_GST_SQE, Put_GST_SQE, Call_JCD_SQE, Put_JCD_SQE,&
   Call_BK_RMSE, Put_BK_RMSE, Call_GST_RMSE, Put_GST_RMSE, Call_JCD_RMSE, Put_JCD_RMSE, &
   Call_GST_incgam_delta1, Call_GST_incgam_delta2, Put_GST_incgam_delta1, Put_GST_incgam_delta2
  REAL(r8), dimension(4)  :: GST_incgam_delta_maxval
  REAL(r8), dimension(2474)  :: VecContractNr, VecSpot, VecStrike, Vectau, Vecbeta, Vecr, Vecq, Vecdelta
  !we need to export files from Excel as Windows formatted text
  OPEN(10,FILE='VecContractNr.txt')
  OPEN(11,FILE='VecSpot.txt')
  OPEN(12,FILE='VecStrike.txt')
  OPEN(13,FILE='Vectau.txt')
```

```fortran
OPEN(14,FILE='Vecbeta.txt')
OPEN(15,FILE='Vecr.txt')
OPEN(16,FILE='Vecq.txt')
OPEN(17,FILE='Vecdelta.txt')
DO i=1, 2474
  READ(10,*) VecContractNr(i);
  READ(11,*) VecSpot(i);
  READ(12,*) VecStrike(i);
  READ(13,*) Vectau(i);
  READ(14,*) Vecbeta(i);
  READ(15,*) Vecr(i);
  READ(16,*) Vecq(i);
  READ(17,*) Vecdelta(i);
ENDDO
DO i=10, 17
  CLOSE(i)
ENDDO
!File to store results
OPEN (unit=iu1,file="test_europeanCEV_FORTRAN.csv",action="write",status="replace")
WRITE (iu1,"('Contract',',','Call (Gamma)',',','Put (Gamma)',',','Call (BK03)',',','Put (BK03)',',','Call (G14)',',','Put (G14)',',','Call (JCD16)',',','Put (JCD16)',',','Call (BK03) AE',',','Put (BK03) AE',',','Call (G14) AE',',','Put (G14) AE',',','Call (JCD) AE',',','Put (JCD) AE',',','Call (BK03) k2',',','Put (BK03) k2',',','Call (G14) k2',',','Put (G14) k2' ,',','Call (JCD) k2',',','Put (JCD) k2',',','Call GST_incgam_delta1',',','Call GST_incgam_delta2',',','Put GST_incgam_delta1',',','Put GST_incgam_delta2':)") !headers
  ! Compute call and Put prices for different methods and store in csv file (test_europeanCEV_FORTRAN.csv)
  ! k2 threshold in $
  k2_threshold = 0.01_r8;
  !CALL cpu_time(start)
  DO j=1, 2474
    ! Knuesel and Bablok (1996) stopping approach to compute cdfgamNC Gamma Series approach
    CALL europeanCEVCall(VecSpot(j),VecStrike(j),Vectau(j),Vecr(j),Vecq(j), Vecdelta(j),Vecbeta(j),3,errtol, .TRUE., EuropeanCallGamma, incgam_delta1, incgam_delta2);
    Call_Gamma(j) = EuropeanCallGamma;
    Call_GST_incgam_delta1(j) = incgam_delta1;
    Call_GST_incgam_delta2(j) = incgam_delta2;
    CALL europeanCEVPut(VecSpot(j),VecStrike(j),Vectau(j),Vecr(j),Vecq(j), Vecdelta(j),Vecbeta(j),3,errtol, .TRUE., EuropeanPutGamma, incgam_delta1, incgam_delta2);
    Put_Gamma(j) = EuropeanPutGamma;
    Put_GST_incgam_delta1(j) = incgam_delta1;
    Put_GST_incgam_delta2(j) = incgam_delta2;
    ! Benton and Krishnamoorthy (2003, Algorithm 7.3)
    CALL europeanCEVCall(VecSpot(j),VecStrike(j),Vectau(j),Vecr(j),Vecq(j), Vecdelta(j),Vecbeta(j),1,errtol, .FALSE., EuropeanCallBK, incgam_delta1, incgam_delta2);
    incgam_delta1 = -1;
    incgam_delta2 = -1;
    Call_BK(j) = EuropeanCallBK;
    !Absolute error
    Call_BK_AE(j) = ABS(EuropeanCallGamma - EuropeanCallBK);
```

```
            !Squared error - to later compute RMSE
            Call_BK_SQE(j) = Call_BK_AE(j) ** 2;
            !k2 considering threshold
            IF (Call_BK_AE(j) > k2_threshold) THEN
              Call_BK_k2(j) = 1.0_r8;
            ELSE
              Call_BK_k2(j) = 0.0_r8;
            END IF
            !Relative error
            Call_BK_RE(j) = Call_BK_AE(j) / Call_Gamma(j)
            CALL europeanCEVPut(VecSpot(j),VecStrike(j),Vectau(j),Vecr(j),Vecq(j), Vecdelta(j),
                 Vecbeta(j),1,errtol, .FALSE., EuropeanPutBK, incgam_delta1, incgam_delta2);
            incgam_delta1 = -1;
            incgam_delta2 = -1;
            Put_BK(j) = EuropeanPutBK;
            !Absolute error
            Put_BK_AE(j) = ABS(EuropeanPutGamma - EuropeanputBK);
            !Squared error - to later compute RMSE
            Put_BK_SQE(j) = Put_BK_AE(j) ** 2;
            !k2 considering threshold
            IF (Put_BK_AE(j) > k2_threshold) THEN
              Put_BK_k2(j) = 1.0_r8;
            ELSE
              Put_BK_k2(j) = 0.0_r8;
            END IF
            !Relative error
            Put_BK_RE(j) = Put_BK_AE(j) / Put_Gamma(j)
            ! Gil, Segura & Temme (2014, GammaCHI package, cdfgamNC)
            CALL europeanCEVCall(VecSpot(j),VecStrike(j),Vectau(j),Vecr(j),Vecq(j), Vecdelta(j)
                 ,Vecbeta(j),2,errtol, .FALSE., EuropeanCallGST, incgam_delta1, incgam_delta2);
            incgam_delta1 = -1;
            incgam_delta2 = -1;
            Call_GST(j) = EuropeanCallGST;
            !Absolute error
            Call_GST_AE(j) = ABS(EuropeanCallGamma - EuropeanCallGST);
            !Squared error - to later compute RMSE
            Call_GST_SQE(j) = Call_GST_AE(j) ** 2;
            !k2 considering threshold
            IF (Call_GST_AE(j) > k2_threshold) THEN
              Call_GST_k2(j) = 1.0_r8;
            ELSE
              Call_GST_k2(j) = 0.0_r8;
            END IF
            !Relative error
            Call_GST_RE(j) = Call_GST_AE(j) / Call_Gamma(j)
            CALL europeanCEVPut(VecSpot(j),VecStrike(j),Vectau(j),Vecr(j),Vecq(j), Vecdelta(j),
                 Vecbeta(j),2,errtol, .FALSE., EuropeanPutGST, incgam_delta1, incgam_delta2);
            incgam_delta1 = -1;
            incgam_delta2 = -1;
            Put_GST(j) = EuropeanPutGST;
            !Absolute error
            Put_GST_AE(j) = ABS(EuropeanPutGamma - EuropeanputGST);
```

```
        !Squared error - to later compute RMSE
        Put_GST_SQE(j) = Put_GST_AE(j) ** 2;
        !k2 considering threshold
        IF (Put_GST_AE(j) > k2_threshold) THEN
          Put_GST_k2(j) = 1.0_r8;
        ELSE
          Put_GST_k2(j) = 0.0_r8;
        END IF
        !Relative error
        Put_GST_RE(j) = Put_GST_AE(j) / Put_Gamma(j)
        ! JCD2016
        CALL europeanCEVCall(VecSpot(j),VecStrike(j),Vectau(j),Vecr(j),Vecq(j), Vecdelta(j)
            ,Vecbeta(j),4,errtol, .FALSE., EuropeanCallJCD, incgam_delta1, incgam_delta2);
        incgam_delta1 = -1;
        incgam_delta2 = -1;
        Call_JCD(j) = EuropeanCallJCD;
        !Absolute error
        Call_JCD_AE(j) = ABS(EuropeanCallGamma - EuropeanCallJCD);
        !Squared error - to later compute RMSE
        Call_JCD_SQE(j) = Call_JCD_AE(j) ** 2;
        !k2 considering threshold
        IF (Call_JCD_AE(j) > k2_threshold) THEN
          Call_JCD_k2(j) = 1.0_r8;
        ELSE
          Call_JCD_k2(j) = 0.0_r8;
        END IF
        !Relative error
        Call_JCD_RE(j) = Call_JCD_AE(j) / Call_Gamma(j)
        CALL europeanCEVPut(VecSpot(j),VecStrike(j),Vectau(j),Vecr(j),Vecq(j), Vecdelta(j),
            Vecbeta(j),4,errtol, .FALSE., EuropeanPutJCD, incgam_delta1, incgam_delta2);
        incgam_delta1 = -1;
        incgam_delta2 = -1;
        Put_JCD(j) = EuropeanPutJCD;
        !Absolute error
        Put_JCD_AE(j) = ABS(EuropeanPutGamma - EuropeanputJCD);
        !Squared error - to later compute RMSE
        Put_JCD_SQE(j) = Put_JCD_AE(j) ** 2;
        !k2 considering threshold
        IF (Put_JCD_AE(j) > k2_threshold) THEN
          Put_JCD_k2(j) = 1.0_r8;
        ELSE
          Put_JCD_k2(j) = 0.0_r8;
        END IF
        !Relative error
        Put_JCD_RE(j) = Put_JCD_AE(j) / Put_Gamma(j)
        WRITE (iu1,"(i4.0,',',f35.30,',',f35.30,',',f35.30,',',f35.30 &
          ,',',f35.30,',',f35.30,',',f35.30,',',f35.30,',',f35.30,',',f35.30,',',f35
              .30,',',f35.30,',',f35.30,',',f35.30, ',',f2.0,',',f2.0,',',f2.0,',',f2
              .0,',',f2.0,',',f2.0,',',es23.16,',',es23.16,',',es23.16,',',es23.16:)") j,
              Call_Gamma(j), Put_gamma(j), Call_BK(j), Put_BK(j), Call_GST(j), Put_GST(j),
              Call_JCD(j), Put_JCD(j), Call_BK_AE(j), Put_BK_AE(j), Call_GST_AE(j),
              Put_GST_AE(j), Call_JCD_AE(j), Put_JCD_AE(j), Call_BK_k2(j), Put_BK_k2(j),
```

```
            Call_GST_k2(j), Put_GST_k2(j), Call_JCD_k2(j), Put_JCD_k2(j),
            Call_GST_incgam_delta1(j), Call_GST_incgam_delta2(j), Put_GST_incgam_delta1(j
            ), Put_GST_incgam_delta2(j)
    ENDDO
    !Max. error (direct computation) for Gil, Segura & Temme (2012, incomplete gamma
        functions ratios P(a,x)
    GST_incgam_delta_maxval(1) = MAXVAL(Call_GST_incgam_delta1);
    GST_incgam_delta_maxval(2) = MAXVAL(Call_GST_incgam_delta2);
    GST_incgam_delta_maxval(3) = MAXVAL(Put_GST_incgam_delta1);
    GST_incgam_delta_maxval(4) = MAXVAL(Put_GST_incgam_delta2);
    ! and Q(a,x))
    PRINT '("Gil, Segura & Temme (2012, incomplete gamma functions ratios P(a,x) and Q(a,
        x)):")'
    PRINT '("Max. error (direct computation) = ",es23.16)' ,MAXVAL(
        GST_incgam_delta_maxval);
    PRINT '("")'
    !Compute MaxAE, MaxRE, RMSE and MeanAE
    PRINT '("MaxAE:")'
    PRINT '("BK Call MaxAE  = ",es23.16)'  ,MAXVAL(Call_BK_AE);
    PRINT '("BK Put MaxAE   = ",es23.16)'  ,MAXVAL(Put_BK_AE);
    PRINT '("GST Call MaxAE = ",es23.16)'  ,MAXVAL(Call_GST_AE);
    PRINT '("GST Put MaxAE  = ",es23.16)'  ,MAXVAL(Put_GST_AE);
    PRINT '("JCD Call MaxAE = ",es23.16)'  ,MAXVAL(Call_JCD_AE);
    PRINT '("JCD Put MaxAE  = ",es23.16)'  ,MAXVAL(Put_JCD_AE);
    PRINT '("")'
    PRINT '("MaxRE:")'
    PRINT '("BK Call MaxRE  = ",es23.16)'  ,MAXVAL(Call_BK_RE);
    PRINT '("BK Put MaxRE   = ",es23.16)'  ,MAXVAL(Put_BK_RE);
    PRINT '("GST Call MaxRE = ",es23.16)'  ,MAXVAL(Call_GST_RE);
    PRINT '("GST Put MaxRE  = ",es23.16)'  ,MAXVAL(Put_GST_RE);
    PRINT '("JCD Call MaxRE = ",es23.16)'  ,MAXVAL(Call_JCD_RE);
    PRINT '("JCD Put MaxRE  = ",es23.16)'  ,MAXVAL(Put_JCD_RE);
    PRINT '("")'
    PRINT '("RMSE:")'
    PRINT '("BK Call RMSE  = ",es23.16)'  ,SQRT(SUM(Call_BK_SQE) / SIZE(Call_BK_SQE));
    PRINT '("BK Put RMSE   = ",es23.16)'  ,SQRT(SUM(Put_BK_SQE) / SIZE(Put_BK_SQE));
    PRINT '("GST Call RMSE = ",es23.16)'  ,SQRT(SUM(Call_GST_SQE) / SIZE(Call_GST_SQE));
    PRINT '("GST Put RMSE  = ",es23.16)'  ,SQRT(SUM(Put_GST_SQE) / SIZE(Put_GST_SQE));
    PRINT '("JCD Call RMSE = ",es23.16)'  ,SQRT(SUM(Call_JCD_SQE) / SIZE(Call_JCD_SQE));
    PRINT '("JCD Put RMSE  = ",es23.16)'  ,SQRT(SUM(Put_JCD_SQE) / SIZE(Put_JCD_SQE));
    PRINT '("")'
    PRINT '("MeanAE:")'
    PRINT '("BK Call MeanAE  = ",es23.16)'  , SUM(Call_BK_AE) / SIZE(Call_BK_AE);
    PRINT '("BK Put MeanAE   = ",es23.16)'  , SUM(Put_BK_AE) / SIZE(Put_BK_AE);
    PRINT '("GST Call MeanAE = ",es23.16)'  , SUM(Call_GST_AE) / SIZE(Call_GST_AE);
    PRINT '("GST Put MeanAE  = ",es23.16)'  , SUM(Put_GST_AE) / SIZE(Put_GST_AE);
    PRINT '("JCD Call MeanAE = ",es23.16)'  , SUM(Call_JCD_AE) / SIZE(Call_JCD_AE);
    PRINT '("JCD Put MeanAE  = ",es23.16)'  , SUM(Put_JCD_AE) / SIZE(Put_JCD_AE);
    PRINT '("")'
END PROGRAM testEuropeanCEV
```

B.2 Code to compute European CEV call option prices

```
MODULE EuropeanCEV_jcd
USE GammaKnueselBablok
USE MarcumFunctionJCD
IMPLICIT NONE
  INTEGER, PARAMETER :: r8 = KIND(0.0d0)
  PRIVATE
  PUBLIC :: europeanCEVCall, europeanCEVPut, bentonKrishF
  CONTAINS

        SUBROUTINE bentonKrishF(w, v, lambda, yy)
                USE IncgamFI

                REAL(r8), INTENT(IN) :: w
            REAL(r8), INTENT(IN) :: v
            REAL(r8), INTENT(IN) :: lambda
            REAL(r8), INTENT(OUT) :: yy
        REAL(r8), PARAMETER :: errtol=1.e-15_r8;  ! demanded accuracy
            REAL(r8) :: x, del, k, a, pp, qq, gamkf, gamkb, poikf, poikb, xtermf, xtermb,
                error, remain, sum
            INTEGER :: maxitr, i, ierr
            LOGICAL :: active
        maxitr = 10000; !maximum number of iterations
        !Set:
        x = 0.5_r8 * w;
        del = 0.5_r8 * lambda;
        k = INT(del);
        a = 0.5_r8 * v + k;
        CALL incgam(a,x,pp,qq,ierr);
        gamkf = pp;
        gamkb = gamkf;
        poikf = EXP(-del + k * LOG(del) - loggam(k + 1.0_r8));
        poikb = poikf;
        xtermf = EXP((a - 1.0_r8) * LOG(x) - x - loggam(a));
        xtermb = xtermf * x / a;
        sum = poikf * gamkf;
        remain = 1.0_r8 - poikf;
        i = 0;
        active = .TRUE.;
        DO WHILE (active .EQV. .TRUE.)
            i = i + 1;
            xtermf = xtermf*x/(a + i - 1.0_r8);
            gamkf = gamkf - xtermf;
            poikf = poikf*del/(k + i);
            sum = sum + poikf*gamkf;
            error = remain*gamkf;
            remain = remain - poikf;
            IF (i > k) THEN
                IF ((error <= errtol) .OR. (i > maxitr)) THEN
                   yy = sum;
                   active = .FALSE.;
```

```
                ENDIF
            ELSE
                xtermb = xtermb*(a - i + 1.0_r8) / x;
                gamkb = gamkb + xtermb;
                poikb = poikb * (k - i + 1.0_r8) / del;
                sum = sum + gamkb * poikb;
                remain = remain - poikb;

                IF ((remain <= errtol ) .OR. (i > maxitr)) THEN
                   yy = sum;
                   active = .FALSE.;
                ENDIF
            ENDIF
        END DO
        END SUBROUTINE bentonKrishF

        SUBROUTINE europeanCEVCall(spot, strike, tau, intrate, divyield, delta, beta,
            DistNC, errtol, check_incgam, &
        z, incgam_delta1, incgam_delta2)
                USE GammaCHI
            IMPLICIT NONE
            REAL(r8), INTENT(IN) :: spot
            REAL(r8), INTENT(IN) :: strike
            REAL(r8), INTENT(IN) :: tau
            REAL(r8), INTENT(IN) :: intrate
            REAL(r8), INTENT(IN) :: divyield
            REAL(r8), INTENT(IN) :: delta
            REAL(r8), INTENT(IN) :: beta
        INTEGER, INTENT(IN) :: DistNC
        REAL(r8), INTENT(IN) :: errtol
        LOGICAL, INTENT(IN) :: check_incgam
            REAL(r8), INTENT(OUT) :: z
        REAL(r8), INTENT(OUT) :: incgam_delta1
        REAL(r8), INTENT(OUT) :: incgam_delta2
        REAL(r8) :: k, x, y, v, cdf1, cdf2, q, p, yy, fvalue, fcvalue, jcd_p, jcd_q
        INTEGER :: ierr
        k = 2.0_r8*(intrate-divyield)/((delta**2.0_r8)*(2.0_r8-beta)*(EXP((intrate-divyield
            )*(2.0_r8-beta)*tau)-1.0_r8));
        x = k * (spot ** (2.0_r8 - beta)) * EXP((intrate - divyield) * (2.0_r8 - beta) *
            tau);
        y = k * strike ** (2.0_r8 - beta);
        v = 1.0_r8 / (2.0_r8 - beta);
        cdf1=0.0_r8;
        cdf2=0.0_r8;
        IF (beta < 2) THEN
            SELECT CASE (DistNC)
                CASE (1)
                    CALL BentonKrishF(2.0_r8 * y, 2.0_r8 + 2.0_r8 * v, 2.0_r8 * x, yy);
                    cdf1 = 1.0_r8 - yy;
                    CALL BentonKrishF(2.0_r8 * x, 2.0_r8 * v, 2.0_r8 * y, yy);
                    cdf2 = yy;
                CASE (2)
```

```
                CALL cdfgamNC(2, 2.0_r8 + 2.0_r8 * v,2.0_r8 * x,2.0_r8 * y, p, q, ierr);
                cdf1 = q;
                CALL cdfgamNC(2, 2.0_r8 * v, 2.0_r8 * y, 2.0_r8 * x, p, q, ierr);
                cdf2 = p;
            CASE (3)
                CALL Gammafunction(1.0_r8 + v, y, x, errtol, check_incgam,fvalue, fcvalue, incgam_delta1)
                cdf1 = fcvalue;
                CALL Gammafunction(v,x,y, errtol, check_incgam,fvalue, fcvalue, incgam_delta2)
                cdf2 = fvalue;
            CASE (4)
                CALL marcumpJCD(y, 1.0_r8 + v, x, jcd_p);
                cdf1 = 1.0_r8 - jcd_p;
                CALL marcumpJCD(x, v, y, jcd_p);
                cdf2 = jcd_p;
        END SELECT
    ENDIF
    IF (beta > 2) THEN
        SELECT CASE (DistNC)
            CASE (1)
                CALL BentonKrishF(2.0_r8 * x, -2.0_r8 * v, 2.0_r8 * y, yy);
                cdf1 = 1.0_r8 - yy;
                CALL BentonKrishF(2.0_r8 * y, 2.0_r8 - 2.0_r8 * v, 2.0_r8 * x, yy);
                cdf2 = yy;
            CASE (2)
                CALL cdfgamNC(2, -2.0_r8 * v, 2.0_r8 * y, 2.0_r8 * x, p, q, ierr);
                cdf1 = q;
                CALL cdfgamNC(2, 2.0_r8 - 2.0_r8 * v, 2.0_r8 * x, 2.0_r8 * y, p, q, ierr);
                cdf2 = p;
            CASE (3)
                CALL Gammafunction(-1.0_r8 * v,x,y, errtol, check_incgam, fvalue, fcvalue, incgam_delta1)
                cdf1 = fcvalue;
                CALL Gammafunction(1.0_r8 - v,y,x, errtol, check_incgam, fvalue, fcvalue, incgam_delta2)
                cdf2 = fvalue;
            CASE (4)
                CALL marcumpJCD(x, -v, y, jcd_p);
                cdf1 = 1.0_r8 - jcd_p;
                CALL marcumpJCD(y, 1.0_r8 - v, x, jcd_p);
                cdf2 = jcd_p;
        END SELECT
    ENDIF
    z = spot * EXP(-divyield * tau) * cdf1 - strike * EXP( -intrate * tau) * cdf2;
    IF (check_incgam .EQV. .FALSE.) THEN
        incgam_delta1 = -1;
        incgam_delta2 = -1;
    END IF
    END SUBROUTINE europeanCEVCall
```

```fortran

SUBROUTINE europeanCEVPut(spot, strike, tau, intrate, divyield, delta, beta, DistNC, errtol, check_incgam, &
 z, incgam_delta1, incgam_delta2)
    USE GammaCHI
    IMPLICIT NONE
    REAL(r8), INTENT(IN) :: spot
    REAL(r8), INTENT(IN) :: strike
    REAL(r8), INTENT(IN) :: tau
    REAL(r8), INTENT(IN) :: intrate
    REAL(r8), INTENT(IN) :: divyield
    REAL(r8), INTENT(IN) :: delta
    REAL(r8), INTENT(IN) :: beta
    INTEGER, INTENT(IN) :: DistNC
    REAL(r8), INTENT(IN) :: errtol
    LOGICAL, INTENT(IN) :: check_incgam
    REAL(r8), INTENT(OUT) :: z
    REAL(r8), INTENT(OUT) :: incgam_delta1
    REAL(r8), INTENT(OUT) :: incgam_delta2
    REAL(r8) :: k, x, y, v, cdf1, cdf2, q, p, yy, fvalue, fcvalue, a, lambda, jcd_p, jcd_q
    INTEGER :: ierr

    k = 2.0_r8*(intrate-divyield)/(delta **2.0_r8*(2.0_r8-beta)*(EXP((intrate-divyield)*(2.0_r8-beta)*tau)-1_r8));
    x = k * (spot ** (2.0_r8 - beta)) * EXP((intrate - divyield) * (2.0_r8 - beta) * tau);
    y = k * strike ** (2.0_r8 - beta);
    v = 1.0_r8 / (2.0_r8 - beta);
    cdf1=0.0_r8;
    cdf2=0.0_r8;
        IF (beta<2) THEN
        SELECT CASE (DistNC)
            CASE (1)
                CALL BentonKrishF(2.0_r8 * y, 2.0_r8 + 2.0_r8 * v, 2.0_r8 * x, yy);
                cdf1 = yy;
                CALL BentonKrishF(2.0_r8 * x, 2.0_r8 * v, 2.0_r8 * y, yy);
                cdf2 = 1.0_r8 - yy;
            CASE (2)
                CALL cdfgamNC(2, 2.0_r8 + 2.0_r8 * v, 2.0_r8 * x, 2.0_r8 * y, p, q, ierr);
                cdf1 = p;
                CALL cdfgamNC(2, 2.0_r8 * v, 2.0_r8 * y, 2.0_r8 * x, p, q, ierr);
                cdf2 = q;
            CASE (3)
                CALL Gammafunction((2.0_r8 + 2.0_r8 * v) / 2.0_r8, (2.0_r8 * y) / 2.0_r8, (2.0_r8 * x) / 2.0_r8, &
                 errtol, check_incgam, fvalue,fcvalue, incgam_delta1)
                cdf1 = fvalue;
                CALL Gammafunction( (2.0_r8 * v) / 2.0_r8, (2.0_r8 * x) / 2.0_r8, (2.0_r8 * y) / 2.0_r8, errtol, &
                    check_incgam, fvalue,fcvalue, incgam_delta2)
```

```
                    cdf2 = fcvalue;
                CASE (4)
                    CALL marcumpJCD(y, 1.0_r8 + v, x, jcd_p);
                    cdf1 = jcd_p;
                    CALL marcumpJCD(x, v, y, jcd_p);
                    cdf2 = 1.0_r8 - jcd_p;
            END SELECT
        END IF

        IF (beta>2) THEN
            SELECT CASE (DistNC)
                CASE (1)
                    CALL BentonKrishF(2.0_r8 * x, -2.0_r8 * v, 2.0_r8 * y, yy);
                    cdf1 = yy;
                    CALL BentonKrishF(2.0_r8 * y, 2.0_r8 - 2.0_r8 * v, 2.0_r8 * x, yy);
                    cdf2 = 1.0_r8 - yy;
                CASE (2)
                    CALL cdfgamNC(2, -2.0_r8 * v, 2.0_r8 * y, 2.0_r8 * x, p, q, ierr);
                    cdf1 = p;
                    CALL cdfgamNC(2, 2.0_r8-2.0_r8 * v, 2.0_r8 * x, 2.0_r8 * y, p, q, ierr)
                        ;
                    cdf2 = q;
                CASE (3)
                    CALL Gammafunction(-1.0_r8 * v,x, y, errtol, check_incgam, fvalue,
                        fcvalue, incgam_delta1)
                    cdf1 = fvalue;
                    CALL Gammafunction(1.0_r8 - v, y, x, errtol, check_incgam, fvalue,
                        fcvalue, incgam_delta2)
                    cdf2 = fcvalue;
                CASE (4)
                    CALL marcumpJCD(x, -v, y, jcd_p);
                    cdf1 = jcd_p;
                    CALL marcumpJCD(y, 1.0_r8 - v, x, jcd_p);
                    cdf2 = 1.0_r8 - jcd_p;
            END SELECT
        ENDIF
        z = - spot * EXP(-divyield * tau) * cdf1 + strike * EXP(-intrate * tau) * cdf2;
        IF (check_incgam .EQV. .FALSE.) THEN
            incgam_delta1 = -1;
            incgam_delta2 = -1;
        END IF
    END SUBROUTINE europeanCEVPut

    REAL FUNCTION Factorial(n)
        IMPLICIT NONE
        INTEGER, INTENT(IN) :: n
        INTEGER :: i
        REAL :: Ans
        Ans = 1
        DO i = 1, n
            Ans = Ans * i
        END DO
```

```
        Factorial = Ans
    END FUNCTION Factorial
END MODULE EuropeanCEV_jcd
```

B.3 Code to compute non-central chi-square function according to Knüsel and Bablok (1996)

```
MODULE GammaKnueselBablok
 USE IncgamFI
 IMPLICIT NONE
 INTEGER, PARAMETER  :: r8 = KIND(0.0d0)
 PRIVATE
 PUBLIC  :: Gammafunction, Ffunction, Fcfunction

 CONTAINS
        SUBROUTINE Gammafunction(a,x,lambda,errtol,check_incgam, fvalue, fcvalue,
            incgam_delta)
        IMPLICIT NONE
            REAL(r8), INTENT(IN) :: a
            REAL(r8), INTENT(IN) :: x
                REAL(r8), INTENT(IN) :: lambda
            REAL(r8), INTENT(IN) :: errtol
            LOGICAL, INTENT(IN) :: check_incgam
            REAL(r8), INTENT(OUT) :: fvalue
            REAL(r8), INTENT(OUT) :: fcvalue
            REAL(r8), INTENT(OUT) :: incgam_delta
                IF (x > (a + lambda)) THEN
                        CALL Fcfunction(a,x,lambda,errtol,fcvalue, check_incgam,
                            incgam_delta);
                        fvalue = 1.0_r8 - fcvalue;
                ELSE
                        CALL Ffunction(a,x,lambda,errtol,fvalue, check_incgam, incgam_delta
                            );
                        fcvalue = 1.0_r8 - fvalue;
                END IF
        END SUBROUTINE Gammafunction

        SUBROUTINE Ifunction(a,x,errtol,ivalue)
        IMPLICIT NONE
            REAL(r8), INTENT(IN) :: a
            REAL(r8), INTENT(IN) :: x
            REAL(r8), INTENT(IN) :: errtol
            REAL(r8), INTENT(OUT) :: ivalue
            REAL(r8) :: fac, mult, b, ib
            INTEGER :: z, n
            !choose n as the smallest positive integer such that fac < errtol
            !fac = 0;
            n = 1;
```

```
            fac = errtol + 1
            DO WHILE (fac > errtol)
                        IF (n == 1) THEN
                                fac = x / a ;
                        ELSE IF (n == 2) THEN
                                fac = x ** 2 / (a * (a + 1));
                        ELSE
                                mult = a * ( a + 1 );
                                DO z = 3 , n
                                        mult = (a + z - 1) * mult;
                                END DO
                                fac = x ** n / (mult);
                        END IF
                n = n + 1;
                END DO
                n = n - 1;
                !backward recursion to compute I
                ib = 0;
                DO z = 1, n
                        b = a + n - z;
                        ib = (x / b) * (1 + ib);
                END DO
                ivalue = ib;
    END SUBROUTINE Ifunction

    SUBROUTINE Jfunction(a,x,errtol,jvalue)
        IMPLICIT NONE
            REAL(r8), INTENT(IN) :: a
            REAL(r8), INTENT(IN) :: x
            REAL(r8), INTENT(IN) :: errtol
            REAL(r8), INTENT(OUT) :: jvalue
            REAL(r8) :: fac, mult, b, jb
            INTEGER :: z, n
                !choose n as the smallest positive integer such that fac < errtol
            !fac = 0;
            n = 1;
            fac = errtol + 1
            DO WHILE (fac > errtol)
                IF (n == 1) THEN
                                fac =  (a - 1) / x;
                        ELSE
                                mult = (a - 1)
                                DO z = 2 , n
                                        mult = mult * (a - n);
                                END DO
                                fac = (mult) / (x ** n);
                        END IF
                n = n + 1;
                END DO
                n = n - 1;
                IF (n > (x + a)) THEN
                END IF
```

```
                !forward recursion to compute J
                jb = 1;
                DO z = 1, (n - 1)
                        b = a - n + z;
                        jb = 1 + ((b / x) * jb);
                END DO
                jvalue = jb;
    END SUBROUTINE Jfunction

    SUBROUTINE Ffunction(a,x,lambda,errtol,fvalue, check_incgam, incgam_delta)
        IMPLICIT NONE
            REAL(r8), INTENT(IN) :: a
            REAL(r8), INTENT(IN) :: x
            REAL(r8), INTENT(IN) :: lambda
            REAL(r8), INTENT(IN) :: errtol
            REAL(r8), INTENT(OUT) :: fvalue
            LOGICAL, INTENT(IN) :: check_incgam
            REAL(r8), INTENT(OUT) :: incgam_delta
            REAL(r8) :: i, p, q, cump, s2,s2_sum , qk1, tk1, tk1_p_1, ival, delta
                INTEGER :: ierr
                REAL(r8), PARAMETER :: eps=0.5e-17_r8;
                !Compute k2 value
                cump = 0.0_r8;
                i = 0.0_r8;
            DO
                CALL incgam(lambda,i,p,q,ierr);
                        IF ((1.0_r8 - p) < errtol) THEN
                                EXIT
                        END IF
                i = i + 1.0_r8;
                END DO
            i = i - 1.0_r8; ! k2 - 1
            s2 = 0.0_r8;
            s2_sum = 0.0_r8;
            DO
                CALL incgam(a + i,x,p,q,ierr);
                !Test for incomplete gamma function accuracy in
                !Q(a+1,x)=Q(a,x)+x^a*exp(-x)/Gamma(a+1) and
                !P(a+1,x)=P(a,x)-x^a*exp(-x)/Gamma(a+1)
                IF (check_incgam .eqv. .TRUE.) THEN
                        incgam_delta=-1;
                        delta = ABS(checkincgam(a + i,x,eps))
                            IF (delta>incgam_delta) THEN
                                        incgam_delta = delta;
                                ENDIF
                        ELSE
                                incgam_delta = -1;
                END IF
                tk1_p_1 = s2; !As we are doing a backward recursion, tk1+1 is the previous
                    s2
                s2 = poisson(i, lambda) * p;
                s2_sum = s2_sum + s2;
```

```
            tk1 = s2; !the new tk1 is the current s2
            qk1 = tk1_p_1 / tk1;
            IF (i == 0.0_r8) THEN
                    EXIT
            END IF
            i = i - 1.0_r8
        END DO
            fvalue = s2_sum;
    END SUBROUTINE Ffunction

    SUBROUTINE Fcfunction(a,x,lambda,errtol,fvalue, check_incgam, incgam_delta)
    IMPLICIT NONE
        REAL(r8), INTENT(IN) :: a
        REAL(r8), INTENT(IN) :: x
        REAL(r8), INTENT(IN) :: lambda
        REAL(r8), INTENT(IN) :: errtol
        REAL(r8), INTENT(OUT) :: fvalue
            LOGICAL, INTENT(IN) :: check_incgam
        REAL(r8), INTENT(OUT) :: incgam_delta
        REAL(r8) :: i, p, q, cump, s2,s2_sum , qk1, tk1, tk1_m_1, jval, delta
            INTEGER :: ierr
            REAL(r8), PARAMETER :: eps=0.5e-17_r8;
            !Compute k1 value
            cump = 0.0_r8;
            i    = 0.0_r8;
        DO
                    CALL incgam(lambda,i,p,q,ierr);
                    IF (p > errtol) THEN
                            EXIT
                    END IF
            i = i + 1.0_r8;
            END DO
            i = i - 1.0_r8;
        s2 = 0.0_r8;
        s2_sum = 0.0_r8;
        DO
            CALL incgam(a + i,x,p,q,ierr);
            !Test for incomplete gamma function accuracy in
            !Q(a+1,x)=Q(a,x)+x^a*exp(-x)/Gamma(a+1) and
            !P(a+1,x)=P(a,x)-x^a*exp(-x)/Gamma(a+1)
            IF (check_incgam .eqv. .TRUE.) THEN
                    incgam_delta=-1;
                    delta = ABS(checkincgam(a + i,x,eps))
                        IF (delta>incgam_delta) THEN
                                incgam_delta = delta;
                            ENDIF
                    ELSE
                            incgam_delta = -1;
            END IF
            tk1_m_1 = s2;
            s2 = poisson(i, lambda) * q;
            s2_sum = s2_sum + s2;
```

```fortran
                    tk1 = s2;
                    qk1 = tk1 / tk1_m_1;
                    IF (qk1 < 1.0_r8) THEN
                            IF ((tk1 / (1.0_r8 - qk1)) <= (errtol * s2_sum)) THEN
                                    EXIT
                            END IF
                        END IF
                    i = i + 1.0_r8
                END DO
                    fvalue = s2_sum;
            END SUBROUTINE Fcfunction

             FUNCTION poisson(i, lambda) RESULT(p)
                    IMPLICIT NONE
                REAL(r8), INTENT(IN) :: i
                REAL(r8), INTENT(IN) :: lambda
                REAL(r8) :: p
                p = EXP(-lambda + i * LOG(lambda) - loggam(i + 1.0_r8));
             END FUNCTION poisson

            FUNCTION pfunction(a, x) RESULT(p)
                    IMPLICIT NONE
                REAL(r8), INTENT(IN) :: a
                REAL(r8), INTENT(IN) :: x
                REAL(r8) :: p
                p = (EXP(- x) * x ** (a - 1.0_r8)) / Gamma(a)
             END FUNCTION pfunction

            RECURSIVE FUNCTION Factorial(n)  RESULT(Fact)
            IMPLICIT NONE
            REAL(r8) :: Fact
            REAL(r8), INTENT(IN) :: n
            IF (n == 0) THEN
               Fact = 1
            ELSE
               Fact = n * Factorial(n-1)
            END IF
            END FUNCTION Factorial
    END MODULE GammaKnueselBablok
```

Appendix C

JDCEV Fortran code

C.1 Code to test the different algorithms to compute JD-CEV option prices

```
PROGRAM testEuropeanJDCEV
USE NuttallFunctionJCD
USE NuttallTilde !JCD2016
USE EuropeanJDCEV
USE NutallF
IMPLICIT NONE
INTEGER, PARAMETER :: r8 = KIND(0.0d0)
REAL(r8) ::  spot, t0, beta, &
p_t0_0_GS, p_t0_D_GS, p_t0_GS, p_t0_0_JCD, p_t0_D_JCD, p_t0_JCD, p_t0_0_GST, p_t0_D_GST,
    p_t0_GST, p_t0_0_S, p_t0_D_S, p_t0_S,finish, start, k2_threshold
REAL(r8), dimension(2500) :: k, T, intrate,divyield, a, b, c
REAL(r8), dimension(2500) :: Put_GS, Put_JCD, Put_GST, Put_JCD_AE, Put_GST_AE, Put_S_AE,
    Put_JCD_k2, Put_GST_k2, Put_JCD_SQE, Put_GST_SQE, Put_JCD_RE, Put_GST_RE
INTEGER :: i, j, nrun
INTEGER, parameter :: iu=20
   OPEN(10,FILE='k.txt')
   OPEN(11,FILE='T.txt')
   OPEN(12,FILE='intrate.txt')
   OPEN(13,FILE='divyield.txt')
   OPEN(14,FILE='a.txt')
   OPEN(15,FILE='b.txt')
   OPEN(16,FILE='c.txt')
   DO i=1, 2500
     READ(10,*) k(i);
     READ(11,*) T(i);
     READ(12,*) intrate(i);
     READ(13,*) divyield(i);
     READ(14,*) a(i);
     READ(15,*) b(i);
     READ(16,*) c(i);
```

```
    ENDDO
    DO i=10, 16
      CLOSE(i)
    ENDDO
open (unit=iu, file="test_EuropeanJDCEV_2.csv", action="write", status="replace")
!write the headers
write (iu,"('Contract Nr',',','P (GS)',',','P (JCD)',',','P (GST)' ,',','P (JCD) AE',',','P (GST) AE' ,',','P (JCD) k2',',','P (GST) k2':)")
!Strike values
 spot = 100.0_r8;
 t0=0;
 beta=-1.0_r8;
! k2 threshold in $
k2_threshold = 0.01_r8;
DO i=1, 2500 !size(K)
    ! Series solution (3.2) in Dias2014a (GS)
    CALL europeanJDCEVPut(spot, K(i), T(i), t0, beta, intrate(i), divyield(i), a(i), b(i), c(i), 1, p_t0_0_GS, p_t0_D_GS, p_t0_GS);
    Put_GS(i) = p_t0_GS;
    ! JCD algorithm in Dias2014a (DN14)
    CALL europeanJDCEVPut(spot, K(i), T(i), t0, beta, intrate(i), divyield(i), a(i), b(i), c(i), 2, p_t0_0_JCD, p_t0_D_JCD, p_t0_JCD);
    Put_JCD(i) = p_t0_JCD;
    Put_JCD_AE(i) = ABS(Put_GS(i) - Put_JCD(i));
    !Squared error - to later compute RMSE
    Put_JCD_SQE(i) = Put_JCD_AE(i) ** 2;
    IF (Put_JCD_AE(i) > k2_threshold) THEN
        Put_JCD_k2(i) = 1.0_r8;
      ELSE
        Put_JCD_k2(i) = 0.0_r8;
    END IF
    !Relative error
    Put_JCD_RE(i) = Put_JCD_AE(i) / Put_GS(i);
    ! Gil2013a NuttallF module (GST13)
    CALL europeanJDCEVPut(spot, K(i), T(i), t0, beta, intrate(i), divyield(i), a(i), b(i), c(i), 3, p_t0_0_GST, p_t0_D_GST, p_t0_GST);
    Put_GST(i) = p_t0_GST;
    Put_GST_AE(i) = ABS(Put_GS(i) - Put_GST(i));
    !Squared error - to later compute RMSE
    Put_GST_SQE(i) = Put_GST_AE(i) ** 2;
    IF (Put_GST_AE(i) > k2_threshold) THEN
        Put_GST_k2(i) = 1.0_r8;
      ELSE
        Put_GST_k2(i) = 0.0_r8;
    END IF
    !Relative error
    Put_GST_RE(i) = Put_GST_AE(i) / Put_GS(i);
    write (iu,"(i4.0,',',es23.16,',',es23.16,',',es23.16,',',es23.16,',',es23.16 ,',',f2.0,',',f2.0:)") i, Put_GS(i), Put_JCD(i), Put_GST(i), Put_JCD_AE(i), Put_GST_AE(i), Put_JCD_k2(i), Put_GST_k2(i)
ENDDO
    !Compute MaxAE, MaxRE, RMSE and MeanAE
```

```
      PRINT '("MaxAE:")'
      PRINT '("JCD Put MaxAE   = ",es23.16)'   ,MAXVAL(Put_JCD_AE);
      PRINT '("GST Put MaxAE  = ",es23.16)'   ,MAXVAL(Put_GST_AE);
      PRINT '("")'

      PRINT '("MaxRE:")'
      PRINT '("JCD Put MaxRE   = ",es23.16)'   ,MAXVAL(Put_JCD_RE);
      PRINT '("GST Put MaxRE  = ",es23.16)'   ,MAXVAL(Put_GST_RE);
      PRINT '("")'

      PRINT '("RMSE:")'
      PRINT '("JCD Put RMSE   = ",es23.16)'   ,SQRT(SUM(Put_JCD_SQE) / SIZE(Put_JCD_SQE));
      PRINT '("GST Put RMSE  = ",es23.16)'   ,SQRT(SUM(Put_GST_SQE) / SIZE(Put_GST_SQE));
      PRINT '("")'

      PRINT '("MeanAE:")'
      PRINT '("JCD Put MeanAE   = ",es23.16)'   , SUM(Put_JCD_AE) / SIZE(Put_JCD_AE);
      PRINT '("GST Put MeanAE  = ",es23.16)'   , SUM(Put_GST_AE) / SIZE(Put_GST_AE);
      PRINT '("")'
END PROGRAM testEuropeanJDCEV
```

C.2 Code to compute European JDCEV put option prices

```
MODULE EuropeanJDCEV
!USE GammaKnueselBablok
IMPLICIT NONE
  INTEGER, PARAMETER  :: r8 = KIND(0.0d0)
  PRIVATE
  PUBLIC  :: europeanJDCEVCall, europeanJDCEVPut
  CONTAINS
    SUBROUTINE europeanJDCEVPut(spot, strike, T, t0, beta, intrate, divyield, a, b, c,
        thetaFunc, p_t0_0, p_t0_D, p_t0)
        USE ThetaFunctionJCD
        USE NuttallTilde
        USE ThetaGammaSeries
        USE NutallF
        IMPLICIT NONE
        REAL(r8), INTENT(IN) :: spot
        REAL(r8), INTENT(IN) :: strike
        REAL(r8), INTENT(IN) :: T
        REAL(r8), INTENT(IN) :: t0
        REAL(r8), INTENT(IN) :: beta
        REAL(r8), INTENT(IN) :: intrate
        REAL(r8), INTENT(IN) :: divyield
        REAL(r8), INTENT(IN) :: a
        REAL(r8), INTENT(IN) :: b
        REAL(r8), INTENT(IN) :: c
        INTEGER, INTENT(IN) :: thetaFunc
        REAL(r8), INTENT(OUT) :: p_t0_0
```

```
        REAL(r8), INTENT(OUT) :: p_t0_D
        REAL(r8), INTENT(OUT) :: p_t0
        REAL(r8) :: x, k, delta_plus, tau, phi1, phi2, phi3, phi1_JCD, phi1_GS ,phi1_GST,
             phi2_JCD, phi2_GS, phi2_GST, phi3_JCD, phi3_GS, phi3_GST, phi3_1, phi3_2, q, q1
             , q2, M1, M2, c_t0
        INTEGER :: ierr
        x=(1.0_r8/ABS(beta))*(spot**ABS(beta));
        k=(1.0_r8/ABS(beta))*(strike**ABS(beta))*EXP(-ABS(beta)*(intrate-divyield+b)*(T-t0)
             );
        delta_plus=((2.0_r8*c+1.0_r8)/(ABS(beta)))+2.0_r8;
        IF ((intrate-divyield+b)==0) THEN
             tau=(a**2.0_r8)*(T-t0);
        ELSE
             tau=(a**2.0_r8/(2.0_r8*ABS(beta)*(intrate-divyield+b)))*(1.0_r8-EXP(-2.0_r8*ABS
                  (beta)*(intrate-divyield+b)*(T-t0)));
        END IF
        !Computation of p_t0_D
        SELECT CASE (thetaFunc)
             CASE (1) ! Series solution (3.2) in Dias2014a
                  CALL thetaGammaSeriesFunction(-1.0_r8/(2.0_r8*ABS(beta)), x**2.0_r8/tau,
                       delta_plus,x**2.0_r8/tau,1,phi3_1);
                  CALL thetaGammaSeriesFunction(-1.0_r8/(2.0_r8*ABS(beta)), x**2.0_r8/tau,
                       delta_plus,x**2.0_r8/tau,-1,phi3_2);
                  phi3_GS = phi3_1 + phi3_2;
                  phi3 = phi3_GS;
             CASE (2) ! JCD algorithm in Dias2014
                  CALL thetaJCD(-1.0_r8/(2.0_r8*ABS(beta)),TINY(0.0_r8)*1000.0_r8, delta_plus
                       ,x**2.0_r8/tau,1,phi3_JCD);
                  phi3 = phi3_JCD;
             CASE (3) ! Gil2013a NuttallF module
                  CALL nuttal(-1.0_r8/(2.0_r8*ABS(beta)), delta_plus/2.0_r8,(x**2.0_r8/tau)
                       /2.0_r8, TINY(0.0_r8)*1000.0_r8, M1, ierr);
                  phi3_GST = M1 * 2**(-1.0_r8/(2.0_r8*ABS(beta)));
                  phi3 = phi3_GST;
        END SELECT
        q=EXP(-b*(T-t0))*((x**2.0_r8/tau)**(1.0_r8/2*ABS(beta)))*phi3;
        p_t0_D=strike*EXP(-intrate*(T-t0))*(1.0_r8-q);
        !Computation of p_t0_0
        SELECT CASE (thetaFunc)
             CASE (1) ! Series solution (3.2) in Dias2014a
                  CALL thetaGammaSeriesFunction(0.0_r8, k**2.0_r8/tau,delta_plus,x**2.0_r8/
                       tau, 1, phi1_GS);
                  CALL thetaGammaSeriesFunction(-1.0_r8/(2.0_r8*ABS(beta)),k**2.0_r8/tau,
                       delta_plus,x**2.0_r8/tau,1, phi2_GS);
                  phi1 = phi1_GS;
                  phi2 = phi2_GS;

             CASE (2) ! JCD algorithm in Dias2014a
                  CALL thetaJCD(0.0_r8, k**2.0_r8/tau,delta_plus,x**2.0_r8/tau, 1, phi1_JCD);
                  CALL thetaJCD(-1.0_r8/(2.0_r8*ABS(beta)),k**2.0_r8/tau, delta_plus,x**2.0
                       _r8/tau,1, phi2_JCD);
                  phi1 = phi1_JCD;
```

```
                phi2 = phi2_JCD;
            CASE (3) ! Gil2013a NuttallF module
                ! Compute Call option price to use theta+ (and its relation with Q_(eta,mu)
                    (x,y) - P_(eta,mu)(x,y) in unavailable for GST).
                ! Use Put-Call parity to compute put option price conditional on no default
                    .
                ! By using this approach we compute two less time a series expansions (M-q
                    above). We use put-call parity instead.
                !phi1+
                CALL nuttal(0.0_r8, delta_plus/2.0_r8, (x**2.0_r8/tau)/2.0_r8, (k**2.0_r8/
                    tau)/2.0_r8,phi1_GST, ierr);
                !phi2+
                CALL nuttal(-1.0_r8/(2.0_r8*ABS(beta)), delta_plus/2.0_r8, (x**2.0_r8/tau)
                    /2.0_r8, (k**2.0_r8/tau)/2.0_r8,phi2_GST, ierr);
                phi1 = phi1_GST;
                phi2 = phi2_GST * (2**(-1.0_r8/(2.0_r8*ABS(beta))));
        END SELECT
        c_t0=EXP(-divyield*(T-t0))*spot*phi1 &
        -EXP(-(intrate+b)*(T-t0))*strike*((x**2.0_r8/tau)**(1.0_r8/(2.0_r8*ABS(beta))))*
            phi2;
        ! put-call parity
        p_t0_0 = c_t0 + strike * EXP(-intrate*(T-t0)) - spot * EXP(-divyield*(T-t0)) -
            p_t0_D;
        !Computation of p_t0
        p_t0=p_t0_0+p_t0_D;
    END SUBROUTINE europeanJDCEVPut
END MODULE EuropeanJDCEV
```

C.3 Code to compute truncated moments benchmark according to the Gamma series approach

```
MODULE ThetaGammaSeries
USE IncgamFI
IMPLICIT NONE
  INTEGER, PARAMETER :: r8 = KIND(0.0d0)
  PRIVATE
  PUBLIC :: thetaGammaSeriesFunction
  CONTAINS
        SUBROUTINE thetaGammaSeriesFunction(p, x, n, lambda, theta, y)
        ! Series solution (3.2) in Dias2014a
        USE GammaCHI
        IMPLICIT NONE
        REAL(r8), INTENT(IN)  :: p
        REAL(r8), INTENT(IN)  :: x
        REAL(r8), INTENT(IN)  :: n
        REAL(r8), INTENT(IN)  :: lambda
        INTEGER , INTENT(IN)  :: theta
        REAL(r8), INTENT(OUT) :: y
```

```
        REAL(r8) :: sum, pp, qq, i, z, z_sum, tk1_m_1,tk1, qk1
        INTEGER :: ierr
        REAL(r8), PARAMETER :: errtol=1.e-15_r8;
        z = 0.0_r8;
        z_sum     = 0.0_r8;
        i         = 0.0_r8;
        DO
            tk1_m_1 = z
            z = (poisson(i, lambda / 2.0_r8) * Ifunction(n/2.0_r8 + i, x / 2.0_r8, p, theta))
            z_sum = z_sum + z;
            tk1 = z;
            qk1 = tk1 / tk1_m_1;
            IF ((tk1 / qk1) > 0.0_r8) THEN
                IF ((tk1 / (qk1)) <= (errtol * z_sum)) THEN
                    EXIT
                END IF
            END IF
            i = i + 1.0_r8;
        END DO
        y = (2.0_r8 ** p) * z_sum;
        END SUBROUTINE thetaGammaSeriesFunction

    FUNCTION poisson(i, lambda) RESULT(p)
        IMPLICIT NONE
        REAL(r8), INTENT(IN) :: i
        REAL(r8), INTENT(IN) :: lambda
        REAL(r8) :: p

        p = EXP(-lambda + i * LOG(lambda) - loggam(i + 1.0_r8));

    END FUNCTION poisson

    FUNCTION Ifunction(a, x, p, theta) RESULT(r)
        IMPLICIT NONE
        REAL(r8), INTENT(IN) :: a
        REAL(r8), INTENT(IN) :: x
        REAL(r8), INTENT(IN) :: p
        INTEGER, INTENT(IN) :: theta
        REAL(r8) :: pp, qq, r
        INTEGER :: ierr
        CALL incgam(a+p,x,pp,qq,ierr);
        IF (theta == -1) THEN
            r = EXP(loggam(a+p) - loggam(a)) * pp;
        ELSE
            r = EXP(loggam(a+p) - loggam(a)) * qq;
        END IF
    END FUNCTION Ifunction

END MODULE ThetaGammaSeries
```