

FAST CALCULATION OF PRICES AND SENSITIVITIES OF EUROPEAN OPTIONS UNDER VARIANCE GAMMA

MARCO DE INNOCENTIS

ABSTRACT. We present new, fast and accurate methods to calculate the prices and sensitivities of European vanilla and digital options under the Variance Gamma model, and the Variance Gamma PDF. For ATM and near-ATM options with short times to maturity, they are usually much faster than methods based on the (inverse) Fourier transform. We show that interpolation near ATM can be very inaccurate, especially for digitals, whereas the results calculated with our method agree with the benchmarks obtained with the most efficient iFT algorithm.

Acknowledgements: The author would like to thank Sergei Levendorskii for the suggestion to consider this problem, and for useful discussions. All errors and omissions are the sole responsibility of the author.

Key words and phrases: European option pricing, Lévy processes, Variance Gamma, VG, KoBoL, CGMY, ATM, inverse Fourier Transform, FFT, flat iFT, parabolic iFT, hyperbolic iFT, ATMVG, FastVG, model calibration.

CONTENTS

1. Introduction	2
2. Option pricing under the VG model	4
2.1. General remarks	4
2.2. A recap of the iFT pricing method	4
3. ATMVG method	5
3.1. Calculation of $\Phi_2(\sigma, \omega; \Lambda)$	7
3.2. Calculation of $\Phi_1(\sigma, \omega; \Lambda)$	10
3.3. Case $\omega \in [-1, 0]$	10
3.4. Algorithm	11
3.5. Numerical results	12
4. Interpolation near ATM	15
4.1. General remarks	15
4.2. Numerical results	15
5. FastVG method	15
5.1. Calculation of $\Psi_2(x; \sigma, \omega; \Lambda)$	17
5.2. Case $\omega \in [-1, 0]$	20

5.3. Algorithm	20
5.4. Digitals and deltas	21
5.5. Numerical results	21
5.6. Vanilla gamma, digital delta and VG PDF	26
6. Conclusions	30
Appendix A. Truncation error bound for the VG option price	30
Appendix B. Adaptive integration methods	32
Appendix C. Comparison of different quadrature methods	32
C.1. ATMVG case	32
C.2. FastVG case	33
Appendix D. Comparison with iFT	33
D.1. Flat iFT	33
D.2. Parabolic iFT	35
D.3. Hyperbolic iFT	35
Appendix E. Interpolation near ATM for vanilla options	37
Appendix F. Deltas, gammas and digitals	41
Appendix G. Numerical calculation of the Hilbert transform	43
Appendix H. Discretization error bound for $\Psi_1(x; \sigma, 0; \Lambda)$.	44
References	45

1. INTRODUCTION

The Variance Gamma (VG) model, introduced to finance by Madan and Seneta in 1990 [19], was the first non-Gaussian Lévy model used in derivative pricing¹, and is available on the Bloomberg system for the pricing and calibration of equity and index options² [7]. Under VG, European options are usually priced by the (inverse) Fourier transform (iFT) method, introduced to finance by Heston [16] and Eydeland [11], and first used in the context of Lévy processes by Boyarchenko and Levendorskii [2, 3, 4] and Carr and Madan [9]. Several variations of the iFT method have been developed over the years [18, 12, 13, 8, 17, 5]. An extensive analysis of this method for Lévy models was first given in [5], which also introduced several new versions and compared their performances with the standard method. As explained in *op. cit.*, all versions of iFT face the greatest difficulties when dealing with options which are either at-the-money (ATM) or very near ATM for processes such as VG or KoBoL (a.k.a. CGMY)

¹The Lévy stable model was introduced to finance by Mandelbrot in 1963 [21], however in its general form it cannot be used for option pricing, since its exponential moments do not exist, therefore $\mathbb{E}[S_t] = \infty$ for all $t > 0$ (cf. Section 2.1).

²The SKEW function on a Bloomberg terminal no longer allows the user to select VG as pricing model. However, the VG calculator is still available on the Bloomberg API.

[3, 6] in the finite variation case³, especially for short times to maturity. As in [5], we use a slightly different definition of ATM from the usual ones, in that we refer to an option with underlying spot S and strike K as ATM if $\ln(S/K) + \mu\tau = 0$, where μ is the (risk-neutral) drift of the process and τ is the time to maturity (see Section 3 for the details). Indeed, the standard iFT implementation (flat iFT) sometimes requires grids with hundreds of thousands of points, and CPU times of the order of a few seconds, to price an ATM option accurately in this situation, which is clearly inadequate for model fitting. For the vanilla delta and gamma, or the price of a digital option, the situation is even worse.

When calibrating to the traded prices of European options, especially those in the foreign exchange (FX) market, there is usually more than one option with very short time to maturity (about one day – two weeks) which is close to ATM. Moreover, these options will generally be among the most liquidly traded, therefore it is important to price them accurately in order to obtain a good parameter fit. During a calibration to market data, the optimization will sometimes come across regions in parameter space such that one of these options is either ATM or very close to it. When iFT techniques are used, and the settings are chosen to keep the absolute error under a specified tolerance, this can result in very long calculation times (cf. [5], Appendix D.1). We develop an alternative method (ATMVG) for the pricing of ATM vanilla and digital options, and the calculation of the vanilla delta, which can replace iFT for the VG model, and show that it is fast, accurate and easy to implement. Using the ATMVG price as a benchmark, we compare the performance of different iFT implementations and confirm the result of [5] that parabolic iFT, introduced in *op. cit.*, is the most efficient one.

However, unless the option's underlying lies within a short distance of ATM, the ATMVG method is of limited practical use, since interpolation near ATM is usually not very accurate for vanillas, and very inaccurate for digitals (cf. Section 4). Therefore we introduce a different method (FastVG) for a fast and accurate calculation of the non-ATM vanilla option price and delta, or the digital option price. Moreover, with only a minor modification, it can also be used to calculate the gamma of a vanilla option, the delta of a digital option, or the VG probability density function (PDF). The latter is particularly useful for historical calibration, e.g. using maximum-likelihood estimation to fit the parameters of the historical process⁴. While for vanilla options FastVG is only significantly quicker than parabolic iFT for short maturities and small distances from ATM, for digital options it is typically much faster for maturities up to about 2-4 weeks, and over a much wider range of strikes⁵.

³In general, the problem occurs with processes of order $\nu \in [0+, 1)$, particularly for ν close to 0. The VG model has $\nu = 0+$.

⁴Such calibrations are often performed in risk management.

⁵This method can also be used for model calibration to the prices of digital options. Liquid digitals are traded in the FX OTC market, and in recent years both the American Stock Exchange

The rest of the paper is organized as follows. In Section 2 we cover some background on option pricing under Lévy processes in general, and the VG model in particular. In Section 3 we introduce the ATMVG method, outline its implementation, and discuss our numerical results. In Section 4, we examine the performance of various interpolation methods near ATM. In Section 5 we introduce the FastVG method for near-ATM option prices and sensitivities, and discuss its performance for OTM options. Section 6 concludes.

2. OPTION PRICING UNDER THE VG MODEL

2.1. General remarks. We assume that the riskless rate $r \geq 0$ and the “dividend yield” $q \geq 0$ are constant, and that, under a risk-neutral measure \mathbb{Q} chosen for the pricing of options on a stock, index, or exchange rate, the underlying follows an exponential Lévy process e^{X_t} . In order for the underlying to be priced, the characteristic exponent ψ of X_t under \mathbb{Q} (definable from $\mathbb{E}^{\mathbb{Q}}[e^{i\xi X_t}] = e^{-t\psi(\xi)}$) must admit the analytic continuation into a strip $\text{Im } \xi \in (\lambda_-, \lambda_+)$ with $\lambda_- \leq -1 < 0 \leq \lambda_+$ and be continuous up to the boundary of the strip [4]. For simplicity, in what follows we will assume that $\lambda_- < -1 < 0 < \lambda_+$. We will also write \mathbb{E} instead of $\mathbb{E}^{\mathbb{Q}}$, with the understanding that all expectations are taken under \mathbb{Q} .

We focus on the case in which the underlying follows a VG process, whose characteristic exponent is given by

$$(2.1) \quad \psi(\xi) = -i\mu\xi + c[\ln(-\lambda_- - i\xi) - \ln(-\lambda_-) + \ln(\lambda_+ + i\xi) - \ln \lambda_+],$$

where c is known as the intensity, λ_{\pm} as the steepness parameters, and μ is the risk-neutral drift, determined from $r - q + \psi(-i) = 0$ [4].

2.2. A recap of the iFT pricing method. The value function $V(\tau, X_t) := V(G; T; t, X_t)$ of an option with payoff at maturity $G(X_T)$ and time to maturity $\tau = T - t$ is given by

$$(2.2) \quad V(\tau, x) = \mathbb{E}[e^{-r\tau} G(X_T) | X_t = x].$$

In what follows, we assume that the Fourier transform \hat{G} of G satisfies the decay condition

$$(2.3) \quad |\hat{G}(\xi)| = O(\xi^{-1}),$$

for $\xi \rightarrow \infty$ along any line $\text{Im } \xi = \omega$, with the exception of a finite number of omegas (this condition is satisfied for both vanillas and digitals). We decompose the payoff G into the Fourier integral

$$(2.4) \quad G(x) = \frac{1}{2\pi} \int_{\text{Im } \xi = \omega} e^{ix\xi} \hat{G}(\xi) d\xi,$$

(AMEX) and the Chicago Board Options Exchange (CBOE) have introduced digital call option contracts.

where $\omega \in (\lambda_-, \lambda_+)$ is such that $e^{\omega x}G(x) \in L^1(\mathbb{R})$. Substituting this into (2.2), we obtain

$$\begin{aligned}
(2.5) \quad V(\tau, x) &= \frac{e^{-r\tau}}{2\pi} \mathbb{E} \left[\int_{\text{Im} \xi = \omega} e^{iX_\tau \xi} \hat{G}(\xi) d\xi \middle| X_t = x \right] \\
&= \frac{e^{-r\tau}}{2\pi} \int_{\text{Im} \xi = \omega} e^{ix\xi} \mathbb{E}[e^{iX_\tau \xi}] \hat{G}(\xi) d\xi \\
&= \frac{e^{-r\tau}}{2\pi} \int_{\text{Im} \xi = \omega} e^{ix\xi - \tau\psi(\xi)} \hat{G}(\xi) d\xi,
\end{aligned}$$

where the standard properties of Lévy processes have been used, and the application of Fubini's theorem is justified due to the regularity assumption (2.3) for G and the fact that, for VG, $e^{-\tau\psi(\xi)}$ decays at infinity as $|\xi|^{-2c\tau}$, where $c > 0$, $\tau > 0$ (cf. Section 3). If we put $\xi = \eta + i\omega$ in (2.5), where ω and η are real, we obtain

$$(2.6) \quad V(\tau, x) = \frac{e^{-r\tau - \omega x'}}{2\pi} \int_{-\infty}^{+\infty} e^{ix'\eta} e^{-\tau\psi_\omega^0(\eta)} \hat{G}_\omega(\eta) d\eta,$$

where $x' = x + \mu\tau$, $\hat{G}_\omega(\eta) = \hat{G}(\eta + i\omega)$, and $\psi_\omega^0(\eta)$ is defined from $\psi(\eta + i\omega) = -i\mu(\eta + i\omega) + \psi_\omega^0(\eta)$. Using the fact that, for real-valued G , we have $\overline{\hat{G}_\omega(\eta)} = \hat{G}(-\eta)$, and that $\overline{\psi_\omega^0(\eta)} = \psi_\omega^0(-\eta)$, we can write (2.6) as

$$(2.7) \quad V(\tau, x) = \frac{e^{-r\tau - \omega x'}}{\pi} \text{Re} \int_0^{+\infty} e^{ix'\eta} e^{-\tau\psi_\omega^0(\eta)} \hat{G}_\omega(\eta) d\eta.$$

In case of the vanilla call and put options with strike K , we have

$$(2.8) \quad \hat{G}(\xi) = -\frac{Ke^{-i\xi \ln K}}{\xi(\xi + i)},$$

where $\text{Im} \xi < -1$ for the call option and $\text{Im} \xi > 0$ for the put. For digitals, see Appendix F.

In what follows, we will redefine $x = \ln(S/K)$, in order to get rid of the oscillating factor in (2.8). Then equation (2.5), for the case of a vanilla call or put option, becomes

$$(2.9) \quad V(\tau, x) = -\frac{Ke^{-r\tau}}{2\pi} \int_{\text{Im} \xi = \omega} \frac{e^{ix\xi - \tau\psi(\xi)}}{\xi(\xi + i)} d\xi.$$

3. ATMVG METHOD

From now on, unless stated otherwise, we will choose ω in (2.9) so that $\omega = (\lambda_+ + \lambda_-)/2$. If $\omega > 0$ we price the put, and if $\omega < -1$ we price the call, applying put-call-parity if needed (we will consider the case $\omega \in [-1, 0]$ in Subsection 3.3). Putting $\sigma = (\lambda_+ - \lambda_-)/2$, we have $\pm\lambda_\pm \pm i\xi = \sigma \pm i\eta$, and therefore

$$\psi_\omega^0(\eta) = -c \ln(-\lambda_- \lambda_+) + c \ln(\sigma^2 + \eta^2).$$

Hence

$$(3.1) \quad V(\tau, x) = -\frac{Ke^{-r'\tau - \omega x'}}{2\pi} \int_{-\infty}^{+\infty} e^{ix'\eta} \frac{(\sigma^2 + \eta^2)^{-c\tau}}{(\eta + i\omega)(\eta + i\omega + i)} d\eta,$$

where $r' = r - c \ln(-\lambda_- \lambda_+)$. We note that, using this parametrization, it is possible to derive an upper bound for the truncation error of the VG option price. The details can be found in Appendix A. Following [5], we refer to the situation in which $x' = 0$ as the ATM case, for both call and put options. Similarly, we define a call (put) option to be out-of-the-money, or OTM, if $x' < 0$ ($x' > 0$). If we set

$$(3.2) \quad I(x'; \sigma, \omega) = \int_{-\infty}^{\infty} e^{ix'\eta} \frac{(\sigma^2 + \eta^2)^{-c\tau}}{(\eta + i\omega)(\eta + i\omega + i)} d\eta,$$

then we can write

$$(3.3) \quad I(0; \sigma, \omega) = \Phi(\sigma, \omega) - \Phi(\sigma, \omega + 1),$$

where

$$(3.4) \quad \Phi(\sigma, \omega) = -i \int_{-\infty}^{+\infty} \frac{(\sigma^2 + \eta^2)^{-c\tau}}{\eta + i\omega} d\eta.$$

Reducing this integral to \mathbb{R}_+ , we obtain

$$\Phi(\sigma, \omega) = 2\omega \int_0^{\infty} \frac{(\sigma^2 + \eta^2)^{-c\tau}}{\omega^2 + \eta^2} d\eta.$$

Hence, if we know how to calculate $\Phi(\sigma, \omega)$, we can obtain the vanilla option price from (3.1), (3.2) and (3.3). We start with the following result.

Proposition 3.1. *For $\eta > \sigma$, the integrand in $\Phi(\sigma, \omega)$ admits the following series expansion*

$$(3.5) \quad (\sigma^2 + \eta^2)^{-c\tau} (\omega^2 + \eta^2)^{-1} = \eta^{-2-2c\tau} \sum_{n=0}^{\infty} c_n(\sigma, \omega, c\tau) \eta^{-2n},$$

where the coefficients $c_n(\sigma, \omega, c\tau)$ can be calculated recursively⁶, by setting $a_0(\sigma, c\tau) = 1$, $c_0(\sigma, \omega, c\tau) = 1$, and, in a cycle w.r.t. $n = 1, 2, \dots$

$$(3.6) \quad a_n(\sigma, c\tau) = -n^{-1}(c\tau + 1 - n)\sigma^2 a_{n-1}(\sigma, c\tau),$$

$$(3.7) \quad c_n(\sigma, \omega, c\tau) = -\omega^2 c_{n-1}(\sigma, \omega, c\tau) + a_n(\sigma, c\tau).$$

⁶Alternatively, a convolution may be used. However, as shown in Appendix C.1, the choice of method makes little difference to the overall CPU time taken for the calculation of the price.

Proof. Follows directly from the binomial theorem. The coefficients $c_n(\sigma, \omega, c\tau)$ are given by

$$c_n(\sigma, \omega, c\tau) = \sum_{j=0}^n a_j(\sigma, c\tau) b_{n-j}(\omega),$$

where in turn

$$(3.8) \quad a_j(\sigma, c\tau) = \frac{c\tau(c\tau + 1) \dots (c\tau + j - 1)}{j!} \sigma^{2j},$$

and $b_\ell(\omega) = \omega^{2\ell}$. From these expressions, (3.6) and (3.7) easily follow. \square

We can break the integral $\Phi(\sigma, \omega)$ into two pieces: $\Phi(\sigma, \omega) = \Phi_1(\sigma, \omega; \Lambda) + \Phi_2(\sigma, \omega; \Lambda)$, where

$$(3.9) \quad \Phi_1(\sigma, \omega; \Lambda) = 2\omega \int_0^\Lambda \frac{(\sigma^2 + \eta^2)^{-c\tau}}{\omega^2 + \eta^2} d\eta,$$

$$(3.10) \quad \Phi_2(\sigma, \omega; \Lambda) = 2\omega \int_\Lambda^\infty \frac{(\sigma^2 + \eta^2)^{-c\tau}}{\omega^2 + \eta^2} d\eta.$$

For $\Lambda > \sigma$, $\Phi_2(\sigma, \omega; \Lambda)$ can be calculated using the expansion (3.5), and $\Phi_1(\sigma, \omega; \Lambda)$, being the integral of a smooth function over a finite interval, can be computed very efficiently using one of several integration techniques. We will examine each of these tasks in detail.

3.1. Calculation of $\Phi_2(\sigma, \omega; \Lambda)$. The next result follows directly from Proposition 3.1.

Proposition 3.2. *The value of $\Phi_2(\sigma, \omega; \Lambda)$, for $\Lambda > \sigma$, is given by*

$$(3.11) \quad \begin{aligned} \Phi_2(\sigma, \omega; \Lambda) &= 2\omega \sum_{n=0}^{\infty} c_n(\sigma, \omega, c\tau) \int_\Lambda^\infty \eta^{-2-2c\tau-2n} d\eta \\ &= 2\omega \sum_{n=0}^{\infty} c_n(\sigma, \omega, c\tau) \frac{\Lambda^{-1-2c\tau-n}}{1+2c\tau+n}. \end{aligned}$$

The next two propositions can be used to determine the number of terms in the series (3.11) which ought to be included in order to reach a desired accuracy.

Proposition 3.3. *Denote by $\Phi_2^{(N)}(\sigma, \omega; \Lambda)$ the approximation to $\Phi_2(\sigma, \omega; \Lambda)$ obtained by discarding all terms from $n = N + 1$ onwards in the expansion (3.11). Then the error of approximation of $\Phi_2(\sigma, \omega; \Lambda)$ by $\Phi_2^{(N)}(\sigma, \omega; \Lambda)$ admits the following bound*

$$(3.12) \quad |\Phi_2^{(N)}(\sigma, \omega; \Lambda) - \Phi_2(\sigma, \omega; \Lambda)| \leq 2|\omega| \varphi_N(\Lambda) \frac{\Lambda^{-3-2c\tau-2N}}{3+2c\tau+2N},$$

where

$$\begin{aligned}
(3.13) \quad \varphi_N(\Lambda) &= |a_{N+1}(\sigma, c\tau)| \frac{\Lambda^{2N+2} - \omega^{2N+2}}{\Lambda^2 - \omega^2} + \omega^{2N+2} \sum_{n=0}^N |a_n(\sigma, c\tau)| \Lambda^{-2n} \\
&+ |a_{N+1}(\sigma, c\tau)| \frac{\omega^{2N+2}}{\Lambda^{2N+2}} + \sum_{n=0}^{N-1} |c_{n+N+1}(\sigma, \omega, c\tau)| \Lambda^{-2n}.
\end{aligned}$$

Proof. Recall that, if $x \in (0, 1)$, $z \in \mathbb{R}$, then the remainder of the series for $f(1+x) := (1+x)^z$ is given by

$$R_N^{(f)} = \sum_{n=N+1}^{\infty} \frac{x^n}{n!} f^{(n)}(1) = \frac{x^{N+1}}{(N+1)!} f^{(N+1)}(1 + \theta x),$$

for some $\theta \in (0, 1)$. Hence

$$R_N^{(f)} \leq \frac{|z(z-1)\dots(z-N)|}{(N+1)!} x^{N+1}.$$

We also note that, if we define $A = \sum_{n=0}^{\infty} A_n$ and $B = \sum_{n=0}^{\infty} B_n$, then the remainder of the product series $C = AB = \sum_{n=0}^{\infty} C_n$ is given by

$$R_N^{(C)} = \sum_{n=N+1}^{\infty} C_n = R_N^{(A)} \sum_{n=0}^N B_n + R_N^{(B)} \sum_{n=0}^N A_n + R_N^{(A)} R_N^{(B)} + \sum_{n=N+1}^{2N} C'_n,$$

where $R_N^{(A)}$ and $R_N^{(B)}$ are the remainders of the series for A and B , respectively, and $C'_n = \sum_{j=n-N}^N A_j B_{n-j}$. Hence

$$(3.14) \quad R_N^{(C)} \leq |R_N^{(A)}| \sum_{n=0}^N |B_n| + |R_N^{(B)}| \sum_{n=0}^N |A_n| + |R_N^{(A)} R_N^{(B)}| + \sum_{n=N+1}^{2N} |C_n|.$$

Then (3.13) follows from

$$\begin{aligned}
\sum_{n=N+1}^{2N} a_n(\sigma, c\tau) \eta^{-2n} &= |a_{N+1}(\sigma, c\tau)| \eta^{-2N-2}, \\
\sum_{n=N+1}^{2N} b_n(\omega) \eta^{-2n} &= |b_{N+1}(\omega)| \eta^{-2N-2} = \frac{\omega^{2N+2}}{\eta^{2N+2}}.
\end{aligned}$$

□

From Proposition 3.3 it is clear that, if ϵ is the tolerance level for the absolute error $|\Phi_2^{(N)}(\sigma, \omega; \Lambda) - \Phi_2(\sigma, \omega; \Lambda)|$, the smallest value of N we can take can be found from

$$(3.15) \quad \frac{\varphi_N(\Lambda)}{(3 + c\tau + 2N)\Lambda^{3+c\tau+2N}} \leq \frac{\epsilon}{2|\omega|}$$

The following iterative procedure can be used to find the smallest value of N , given the error tolerance ϵ : in a cycle w.r.t. $N = 1, 2, \dots$, calculate the LHS of (3.15), and stop as soon as its value drops below that of the RHS.

If $c\tau \leq 1$, as is often the case for short times to maturity and typical parameter ranges, then the next result can be used instead of Proposition 3.3.

Proposition 3.4. *If $c\tau \leq 1$, the error of approximation of $\Phi_2(\sigma, \omega; \Lambda)$ by $\Phi_2^{(N)}(\sigma, \omega; \Lambda)$ admits the following bound*

$$(3.16) \quad |\Phi_2^{(N)}(\sigma, \omega; \Lambda) - \Phi_2(\sigma, \omega; \Lambda)| \leq 2|\omega| \tilde{\varphi}_N(\Lambda) \frac{\Lambda^{-3-2c\tau-2N}}{3+2c\tau+2N},$$

where

$$(3.17) \quad \begin{aligned} \tilde{\varphi}_N(\Lambda) = & C(N+1) \frac{\sigma^{2N+2} \Lambda^{2N+2} - \omega^{2N+2}}{\Lambda^{2N}} \frac{\Lambda^{2N+2} - \omega^{2N+2}}{\Lambda^2 - \omega^2} + D(N) \frac{\omega^{2N+2} \Lambda^{2N+2} - \sigma^{2N+2}}{\Lambda^{2N}} \frac{\Lambda^{2N+2} - \sigma^{2N+2}}{\Lambda^2 - \sigma^2}, \\ & + C(N+1) \frac{\omega^{2N+2}}{\Lambda^{2N+2}} + \frac{D(N-1)}{1 - \frac{\sigma^2}{\omega^2}} \left(\frac{1 - \omega^{2N}}{1 - \omega^2} - \frac{1 - \sigma^{2N}}{1 - \sigma^2} \right) \Lambda^{2-2N}, \end{aligned}$$

and

$$(3.18) \quad C(n) = \min[1, (c\tau(1 + c\tau(n-1)))],$$

$$(3.19) \quad D(n) = \min(1, c\tau(1 + nc\tau)).$$

Proof. If $c\tau \leq 1$, then we have

$$(3.20) \quad \frac{|c\tau(c\tau+1) \dots (c\tau+n-1)|}{n!} \leq C(n).$$

For the case $C(n) = 1$, this follows from

$$\frac{|c\tau(c\tau+1) \dots (c\tau+n-1)|}{n!} \leq \frac{1 \cdot 2 \cdot \dots \cdot n}{n!} = 1.$$

For the more general case, note that the LHS of (3.20) can be re-written as $c\tau n^{-1} P_{n-1}$, where

$$P_n = \begin{cases} 1 & n = 0, \\ \frac{(1+c\tau)(2+c\tau) \dots (n+c\tau)}{n!} = \prod_{k=1}^n \left(1 + \frac{c\tau}{k}\right) & n > 0. \end{cases}$$

It can be shown by induction that $P_n \leq 1 + nc\tau$, from which (3.20) follows. In a similar way, one can prove the following result. For $\zeta \in \mathbb{R}$, we have

$$(3.21) \quad \sum_{n=0}^N \frac{|c\tau(c\tau+1) \dots (c\tau+n-1)|}{n!} \zeta^{2n} \leq D(N) \frac{1 - \zeta^{2N+2}}{1 - \zeta^2}.$$

Then (3.17) follows from (3.13) and (3.20)–(3.21). \square

3.2. Calculation of $\Phi_1(\sigma, \omega; \Lambda)$. As mentioned earlier, $\Phi_1(\sigma, \omega; \Lambda)$ can be computed to a very high degree of accuracy using one of several numerical integration techniques. In Appendix B we briefly review the application of adaptive integration schemes for this type of problem. Overall, the best results were obtained by Clenshaw-Curtis integration and MATLAB's implementation of vectorized Gauss-Kronrod quadrature.

3.3. Case $\omega \in [-1, 0]$. In this subsection, we will use $V(\tau, x; \omega)$ to denote the RHS of (2.9) for a specific choice of ω .

3.3.1. Case $\omega \in (-1, 0)$. Define $V_c(\tau, x)$ and $V_p(\tau, x)$ to be the vanilla call and put prices, respectively, and set $V_{(-1,0)}(\tau, x) = V(\tau, x; \omega)$ for $\omega \in (-1, 0)$. Then, using a similar procedure to that in [4, §4.2.2] to give a proof of the well-known vanilla put-call parity relationship

$$(3.22) \quad V_c(\tau, x) - V_p(\tau, x) = e^{-q\tau} K e^x - e^{-r\tau} K,$$

we can express the call price in terms of $V_{(-1,0)}(\tau, x)$.

Proposition 3.5. *The following relationship holds between $V_c(\tau, x)$ and $V_{(-1,0)}(\tau, x)$*

$$V_c(\tau, x) - V_{(-1,0)}(\tau, x) = e^{-q\tau} K e^x.$$

Proof. We have

$$V_c(\tau, x) - V_{(-1,0)}(\tau, x) = \left(\int_{-\infty+i\omega_-}^{\infty+i\omega_-} - \int_{-\infty+i\sigma}^{\infty+i\sigma} \right) \frac{-K e^{-r\tau} e^{ix\xi - \tau\psi(\xi)}}{2\pi \xi(\xi+i)} d\xi,$$

where $\omega_- < -1$, $\sigma \in (-1, 0)$. The integrand has only one singularity in the strip $\text{Im } \xi \in [\omega_-, \sigma]$, a simple pole at $\xi = -i$. From this, the result easily follows. \square

3.3.2. Case $\omega \in \{-1, 0\}$. Define $V_{-1}(\tau, x) = V(\tau, x; -1)$ and $V_0(\tau, x) = V(\tau, x; 0)$, with the corresponding integrals in (2.9) defined in the sense of Cauchy's principal value. The next two propositions show how to express the call price in terms of $V_{-1}(\tau, x)$ and $V_0(\tau, x)$.

Proposition 3.6. *The following relationship holds between $V_c(\tau, x)$ and $V_{-1}(\tau, x)$*

$$V_c(\tau, x) - V_{-1}(\tau, x) = \frac{1}{2} e^{-q\tau} K e^x.$$

Proof. We have

$$V_c(\tau, x) - V_{-1}(\tau, x) = \left(\int_{-\infty+i\omega_-}^{\infty+i\omega_-} - P \int_{-\infty-i}^{\infty-i} \right) \frac{-K e^{-r\tau} e^{ix\xi - \tau\psi(\xi)}}{2\pi \xi(\xi+i)} d\xi,$$

where P denotes the Cauchy principal value of the integral. Compared to the situation in Proposition 3.5, we now have only half the contribution at the simple pole $\xi = -i$. \square

Proposition 3.7. *The following relationship holds between $V_c(\tau, x)$ and $V_0(\tau, x)$*

$$V_c(\tau, x) - V_0(\tau, x) = e^{-q\tau} K e^x - \frac{1}{2} e^{-r\tau} K.$$

Proof. We have

$$V_c(\tau, x) - V_0(\tau, x) = \left(\int_{-\infty+i\omega_-}^{\infty+i\omega_-} -P \int_{-\infty}^{\infty} \right) \frac{-K e^{-r\tau} e^{ix\xi - \tau\psi(\xi)}}{2\pi \xi(\xi + i)} d\xi,$$

Compared to the situation with put-call-parity [4, §4.2.2], we now have only half the contribution at the simple pole $\xi = 0$, but the full contribution at the simple pole $\xi = -i$. \square

Finally, we note that, at ATM, we have

$$\begin{aligned} V_{-1}(\tau, -\mu\tau) &= -\frac{K e^{-r'\tau}}{2\pi} \Phi(\sigma, -1), \\ V_0(\tau, -\mu\tau) &= \frac{K e^{-r'\tau}}{2\pi} \Phi(\sigma, 1). \end{aligned}$$

3.4. Algorithm. From the results of the previous sections, the following procedure can be used to calculate the ATM price of a vanilla option under VG. The simple modifications for the case of the ATM vanilla delta and digital price are left to the reader (cf. Appendix F).

1. Fix a value of Λ . As noted earlier, and as will be seen from the numerical results in the next section, this choice is largely irrelevant to the accuracy which can be achieved by the algorithm. The only requirement is that $\Lambda > \sigma$. For a fast implementation, we recommend the choice $\Lambda = 1.5\sigma$, since it is neither so small that it would force one to include several additional terms in the asymptotic expansion of $\Phi_2(\sigma, \omega; \Lambda)$, nor so large as to significantly slow down the calculation of $\Phi_1(\sigma, \omega; \Lambda)$.
2. Fix the value of the error tolerance ϵ for the option price, broken down as $\epsilon = \epsilon_1 + \epsilon_2$, where ϵ_1 is used for the calculation involving $\Phi_1(\sigma, \omega; \Lambda)$ and ϵ_2 for that involving $\Phi_2(\sigma, \omega; \Lambda)$. One may set $\epsilon_1 = \epsilon_2 = \epsilon/2$; however, as we will see in Subsection 3.5, such a choice may not give the best performance, and in practice it is usually safe to set ϵ_1 about four orders of magnitude larger than ϵ , and ϵ_2 about one order of magnitude larger than ϵ . Set $\epsilon'_1 = \epsilon_1 2\pi K^{-1} e^{r'\tau + \omega x'}$, and $\epsilon'_2 = \epsilon_2 2\pi K^{-1} e^{r'\tau + \omega x'}$.
3. Calculate $\Phi_1(\sigma, \omega; \Lambda)$ and $\Phi_1(\sigma, \omega + 1; \Lambda)$ (cf. definition (3.9)), e.g. by using one of the numerical integration methods discussed in Appendix B, with error tolerance $\epsilon'_1/2$.
4. Calculate $\Phi_2(\sigma, \omega; \Lambda)$ and $\Phi_2(\sigma, \omega + 1; \Lambda)$ (cf. definition (3.10)) by the method of Subsection 3.1, i.e. using the asymptotic expansion (3.11) with N terms, where N can be determined either from Proposition 3.4, if $c\tau \leq 1$, or from Proposition 3.3, if $c\tau > 1$. In either case, use error tolerance $\epsilon'_2/2$.

5. Calculate the price by using (3.1)–(3.3) and (3.9)–(3.10), applying put-call-parity and/or one of Propositions 3.5–3.7, if needed.

3.5. Numerical results.

3.5.1. *Choice of Λ .* First, we look at the behaviour of the ATMVG price for varying truncation parameter Λ . As pointed out in Subsection 3.4, the choice of Λ ought to be largely irrelevant to the accuracy of the algorithm on theoretical grounds, and should at most affect the overall CPU time taken for the calculation⁷. We check this by comparing the ATMVG prices and CPU times for a wide range of Λ . Throughout this Section, we will consider the case of an ATM call option with $K = 1$, $\tau = 0.004$ (approximately equal to one trading day), $r = 0.03$, $q = 0$, under process parameters⁸ $\lambda_- = -11$, $\lambda_+ = 8$, and $m_2 = 0.16$. We use the adaptive Gauss-Lobatto rule for the calculation of $\Phi_1(\sigma, \omega; \Lambda)$. As we will see, other methods are faster, but adaptive Gauss-Lobatto quadrature can cope more easily with large ranges of integration and low error tolerances. Therefore, all the plots of CPU times in the remainder of this section should be used only to judge the *relative* performance of the algorithm for different choices of truncation parameter and error tolerances (the comparison of CPU times corresponding to other integration methods can be found in Appendix C.1). We take error tolerances $\epsilon_1 = \epsilon_2 = 10^{-16}$, include twice as many terms in the series (3.11) as necessary, and refer to the price calculated in this way as our “benchmark”. For a given choice of Λ we denote it by $V_{\text{bm}}(\Lambda)$. In Figure 1a we plot the absolute difference between $V_{\text{bm}}(\Lambda)$ and $V_{\text{bm}}(\Lambda_0)$, where $\Lambda_0 = 1.2\sigma$ and Λ varies from Λ_0 to 10^7 . As can be clearly seen from the picture, the differences are of the order of floating point relative accuracy⁹ or smaller. The corresponding CPU times, shown in Figure 1b, range from 40.81 ms for $\Lambda = \Lambda_0$ to 145.63 ms for $\Lambda = 10^7$. We conclude that the choice of Λ is indeed irrelevant for the accuracy of the price, and from now on simply set $\Lambda = 1.5\sigma$, unless stated otherwise.

3.5.2. *Choice of ϵ_1 and ϵ_2 .* We consider the effect of the error tolerance on the calculation of $\Phi_1(\sigma, \omega; \Lambda)$ and $\Phi_2(\sigma, \omega; \Lambda)$. In Figure 2 we plot the absolute difference in the ATMVG price for varying ϵ_1 w.r.t. $\epsilon_1 = 10^{-16}$, while ϵ_2 is held fixed at 10^{-16} , and the corresponding CPU times. In Figure 3 we do the same for varying ϵ_2 , while ϵ_1 is held fixed at 10^{-16} . We note that

1. The price does not change at all between $\epsilon_1 = 10^{-16}$ and $\epsilon_1 \simeq 2 \cdot 10^{-11}$. Even for larger values of ϵ_1 , the error is still several orders of magnitude smaller than ϵ_1 , and is of the order of 10^{-15} for $\epsilon_1 = 10^{-8}$. This remarkable accuracy is due to the

⁷The calculations were carried out in MATLAB 7.9.0, on a PC with Intel Core 2 Duo, CPU P8400, 2.26 GHz and 3 GB RAM, running Windows XP Professional.

⁸ m_2 denotes the instantaneous second moment, which determines c via $c = m_2(\lambda_-^{-2} + \lambda_+^{-2})^{-1}$.

⁹Equal to $2^{-52} \simeq 2.22 \cdot 10^{-16}$.

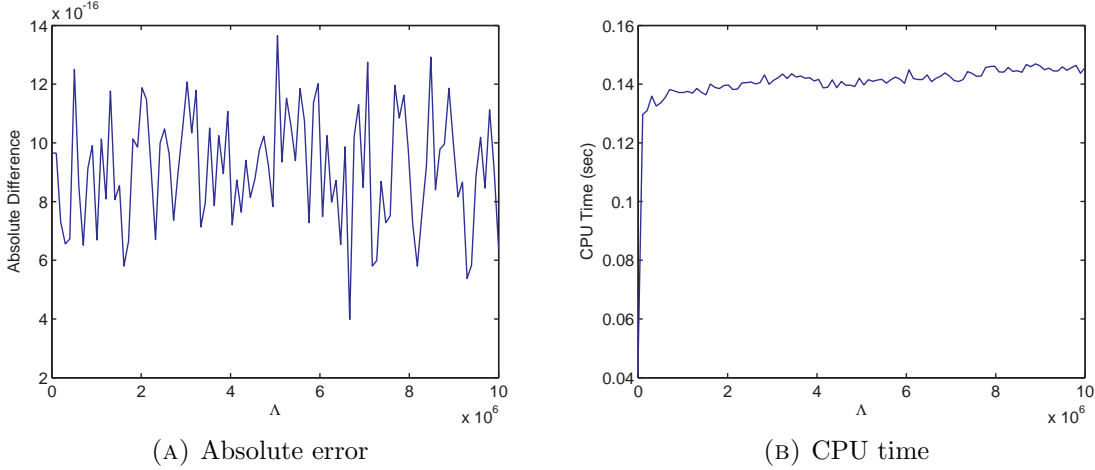


FIGURE 1. $|V_{\text{bm}}(\Lambda) - V_{\text{bm}}(\Lambda_0)|$, and corresponding CPU times, for varying Λ , and $\Lambda_0 = 1.5\sigma$, for an ATM vanilla call option with $K = 1$, $\lambda_- = -11$, $\lambda_+ = 8$, $m_2 = 0.16$, $r = 0.03$, $q = 0$, $\tau = 0.004$, for $\epsilon_1 = \epsilon_2 = 10^{-16}$.

high efficiency of adaptive quadrature methods for smooth integrands and small intervals.

2. The CPU time falls almost monotonically for decreasing ϵ_1 , from 48.77 ms for $\epsilon_1 = 10^{-16}$ to 4.54 ms for $\epsilon_1 = 10^{-8}$.
3. As ϵ_2 increases from 10^{-16} to 10^{-8} , the error increases more or less monotonically, staying about two orders of magnitude below ϵ_2 .
4. The CPU time is largely independent of ϵ_2 , reflecting the fact that the calculation of $\Phi_2(\sigma, \omega; \Lambda)$ is very fast compared to that of $\Phi_1(\sigma, \omega; \Lambda)$, as shown in Appendix C.1.

In light of these results, even if accuracy of the order of floating point error is desired for the benchmark, one might as well take ϵ_1 around 10^{-12} or even slightly larger, while ϵ_2 can be chosen in the range 10^{-16} – 10^{-15} .

3.5.3. *Comparison with iFT.* The absolute difference between the ATMVG benchmark and the price calculated by parabolic iFT, with $\alpha = 3$, using a very long and fine grid ($\zeta = 0.1$, $\Lambda = 100000$) is very small, at $2.56 \cdot 10^{-15}$. In Appendix D we include a detailed comparison of the ATMVG prices with those obtained by flat iFT and the new variants introduced in [5]: parabolic and hyperbolic iFT. The results confirm the analysis of *op. cit.*, according to which parabolic iFT is by far the most efficient version of the method, both in terms of accuracy and CPU time.

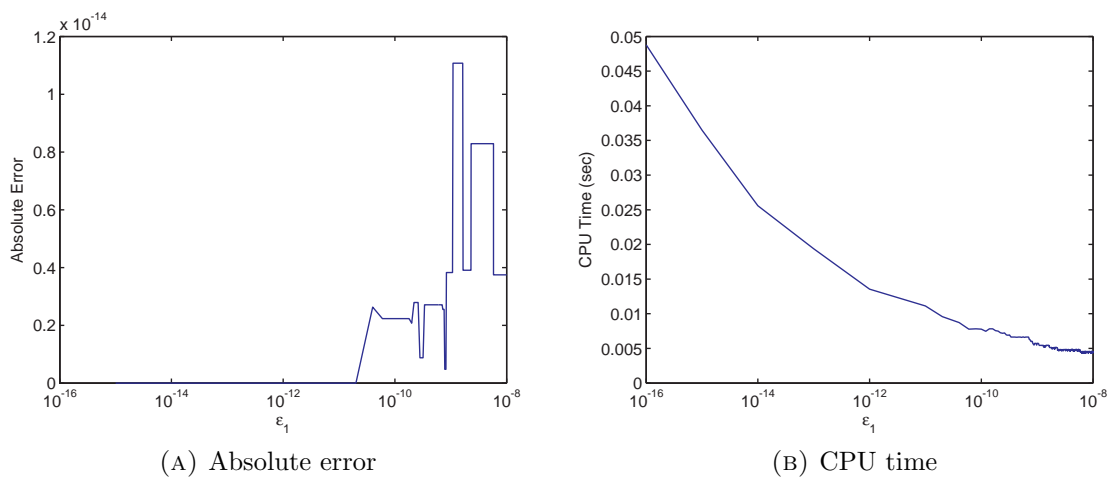


FIGURE 2. Absolute errors in the ATMVG price, and corresponding CPU times, for varying error tolerance level ϵ_1 w.r.t. $\epsilon = 10^{-16}$, for an ATM vanilla call option with $K = 1$, $\lambda_- = -11$, $\lambda_+ = 8$, $m_2 = 0.16$, $r = 0.03$, $q = 0$, $\tau = 0.004$, for $\Lambda = 1.5\sigma$.

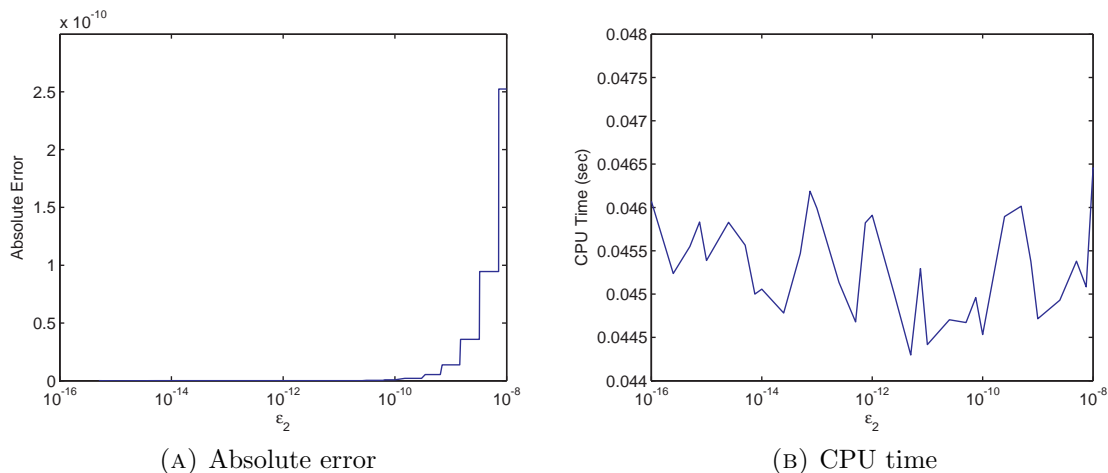


FIGURE 3. Absolute errors in the ATMVG price, and corresponding CPU times, for varying error tolerance level ϵ_2 w.r.t. $\epsilon = 10^{-16}$, for an ATM vanilla call option with $K = 1$, $\lambda_- = -11$, $\lambda_+ = 8$, $m_2 = 0.16$, $r = 0.03$, $q = 0$, $\tau = 0.004$, for $\Lambda = 1.5\sigma$.

4. INTERPOLATION NEAR ATM

4.1. General remarks. In practice, when the option prices for a set of different strikes are needed, one calculates the prices on a regular grid of log-strikes¹⁰, and then interpolates. Normally, linear interpolation is used. A situation which occurs very often during model fitting is that, at some stages of the calibration, one or more options will be very close to, but not exactly ATM. If one of the option maturities is very short¹¹, this can considerably slow down the calibration process. An obvious alternative is therefore to choose the x -grid so that all points will be far from ATM, at least for small time to maturity and moderate drift, and interpolate with the ATMVG price. In this section, we study the performance of different interpolation methods near the ATM point.

4.2. Numerical results. Take $\Delta \gg \delta > 0$, $M = 32$, $x_j = -\mu\tau - j\Delta$, $j = 0, \dots, M$. In Appendix E, we compare the performance of different interpolation methods to approximate the price of a vanilla call option at $x = -\mu\tau - \delta$, from the prices alone, or the prices and deltas, of vanilla call options with maturity $\tau = 0.004$ (approximately one day) on the grid $(x_j)_{j=0}^{N-1}$, for $N = 1, \dots, M$, for $\Delta = 0.02$, $\delta = 0.001$, $M = 32$. The results show that near ATM the relative errors of linear interpolation are several orders of magnitude larger than far from ATM. By using polynomial interpolation of higher order, and taking into account the option's deltas as well, it is usually possible to achieve somewhat higher accuracies, but for small maturities one seldom attains relative errors smaller than 0.5-1%, except in the case of very large steepness parameters. The price of a digital option is given by a similar expression to that of the vanilla delta (compare (F.1)-(F.3) with (F.4)-(F.6)). Due to the irregularity of the latter under VG (and in general for small $\nu > 0$), and the poor results obtained by interpolating on both the ATM and OTM deltas (cf. Figure 17b in Appendix E), we expect interpolation to produce very large errors for digitals, and it does. Figure 4 shows logarithmic plots of the interpolation errors at varying x' , for vanilla and digital call options with $\tau = 1/26$ (approximately two weeks). From these pictures it can be seen that, even for a longer time to maturity than one day, the interpolation errors near ATM can be very large, particularly for digitals.

5. FASTVG METHOD

As we have seen, interpolation near the ATM point can be very inaccurate, even when taking account of both prices and deltas at several points, especially for digitals. Since near-ATM options tend to be the most liquid, it is important for model fitting purposes to calculate their prices accurately. In this section, we outline a method (FastVG) similar to that of Section 3, but valid for $x' \neq 0$. This can be used as an

¹⁰As noted in [5], calculating the option price as a function of log-strike rather than log-spot amounts to performing a trivial change of variable in (2.9).

¹¹This is usually the case in the FX market, where one-day maturity options can be very liquid.

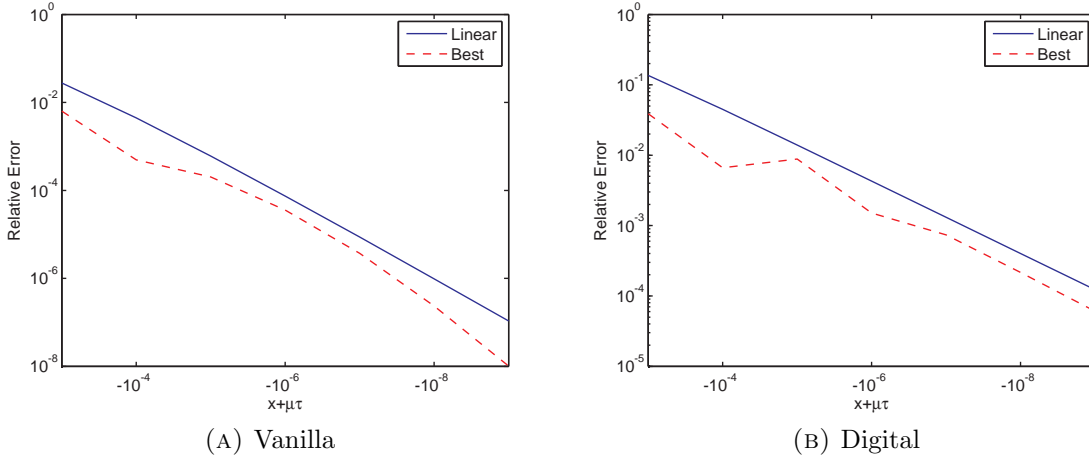


FIGURE 4. Interpolation errors near ATM, at varying $x' = x + \mu\tau$, for a vanilla and a digital call option with $K = 1$, $\lambda_- = -11$, $\lambda_+ = 8$, $m_2 = 0.16$, $r = 0.03$, $q = 0$, $\tau = 1/26$. We compare the accuracy of linear interpolation with the best value obtained by polynomial interpolation through the points $x'_j = -j \cdot 0.02$, $j = 0, \dots, M - 1$, $M = 2, \dots, 32$ (with or without deltas).

alternative to iFT and integration-along-cut (IAC) [4, 17] for the case in which x' is close to, but not quite zero, when iFT can take a long time to produce accurate results [5]. As in Section 3, we consider the vanilla option case when developing the algorithm. For the extension to the case of digitals, see Appendix F.

Taking (2.7) into account, the vanilla option price is given by

$$(5.1) \quad V(\tau, x) = -\frac{K}{\pi} e^{-r'\tau - \omega x'} \operatorname{Re} I^+(x'; \sigma, \omega),$$

where

$$(5.2) \quad I^+(x'; \sigma, \omega) = \int_0^\infty e^{ix'\eta} \frac{(\sigma^2 + \eta^2)^{-c\tau}}{(\eta + i\omega)(\eta + i\omega + i)} d\eta,$$

for $x' \neq 0$. In a similar way to (3.3), we have

$$(5.3) \quad I^+(x; \sigma, \omega) = \Psi(x; \sigma, \omega) - \Psi(x; \sigma, \omega + 1),$$

where

$$(5.4) \quad \Psi(x; \sigma, \omega) = -i \int_0^\infty e^{ix\eta} \frac{(\sigma^2 + \eta^2)^{-c\tau}}{\eta + i\omega} d\eta.$$

For $\Lambda > 0$, we set

$$(5.5) \quad \Psi_1(x; \sigma, \omega; \Lambda) = -i \int_0^\Lambda e^{ix\eta} \frac{(\sigma^2 + \eta^2)^{-c\tau}}{\eta + i\omega} d\eta,$$

$$(5.6) \quad \Psi_2(x; \sigma, \omega; \Lambda) = -i \int_{\Lambda}^{\infty} e^{ix\eta} \frac{(\sigma^2 + \eta^2)^{-c\tau}}{\eta + i\omega} d\eta,$$

so that $\Psi(x; \sigma, \omega) = \Psi_1(x; \sigma, \omega; \Lambda) + \Psi_2(x; \sigma, \omega; \Lambda)$. As in the case of $\Phi_1(\sigma, \omega)$, $\Psi_1(x; \sigma, \omega; \Lambda)$ can be calculated extremely accurately using one of several integration methods (cf. Appendix B).

5.1. Calculation of $\Psi_2(x; \sigma, \omega; \Lambda)$. The following result corresponds to Proposition 3.2 for the ATM case.

Proposition 5.1. *For $x \neq 0$ and $\Lambda > \sigma$ we have*

$$(5.7) \quad \Psi_2(x; \sigma, \omega; \Lambda) = -i \sum_{n=0}^{\infty} c_n(\sigma, \omega, c\tau) (-ix)^{2c\tau+n} \Gamma(-2c\tau - n, -ix\Lambda),$$

where $\Gamma(\cdot, \cdot)$ denotes the principal branch of the upper incomplete gamma function, defined by

$$\Gamma(s, x) = \int_x^{\infty} e^{-t} t^{s-1} dt,$$

where $\operatorname{Re} s > 0$, for integration paths which do not include the origin or cross the negative real axis [1]. The coefficients $c_n(\sigma, \omega, c\tau)$ can be calculated by setting $a_0(\sigma, c\tau) = 1$, $c_0(\sigma, \omega, c\tau) = 1$, and using the following recurrence relations

$$(5.8) \quad a_n(\sigma, c\tau) = -\frac{2\sigma^2}{n} \left(c + \frac{n}{2} - 1 \right),$$

$$(5.9) \quad c_n(\sigma, \omega, c\tau) = a_n(\sigma, c\tau) - i\omega c_{n-1}(\sigma, c\tau).$$

Proof. For $|\eta| \geq \Lambda > \sigma$, we can write¹²

$$(5.10) \quad \begin{aligned} \frac{(\sigma^2 + \eta^2)^{-c\tau}}{\eta + i\omega} &= \eta^{-1-2c\tau} \left(1 + \frac{\sigma^2}{\eta^2} \right)^{-c\tau} \left(1 + \frac{i\omega}{\eta} \right)^{-1} \\ &= \eta^{-1-2c\tau} \left(\sum_{j=0}^{\infty} \alpha_j(\sigma, c\tau) \eta^{-2j} \right) \left(\sum_{k=0}^{\infty} b_k(\omega) \eta^{-k} \right) \\ &= \eta^{-1-2c\tau} \left(\sum_{j=0}^{\infty} a_j(\sigma, c\tau) \eta^{-j} \right) \left(\sum_{k=0}^{\infty} b_k(\omega) \eta^{-k} \right) \\ &= \eta^{-1-2c\tau} \sum_{n=0}^{\infty} c_n(\sigma, \omega, c\tau) \eta^{-n}, \end{aligned}$$

¹²With a slight abuse of notation, we re-use some of the symbols from Section 3 for the series expansion coefficients.

where

$$(5.11) \quad \alpha_j(\sigma, c\tau) = (-1)^j \frac{c\tau(c\tau+1)\dots(c\tau+j-1)}{j!} \sigma^{2j},$$

$$(5.12) \quad a_j(\sigma, c\tau) = (-1)^{j/2} \frac{c\tau(c\tau+1)\dots(c\tau+j/2-1)}{(j/2)!} \sigma^j \frac{(-1)^j + 1}{2},$$

$$(5.13) \quad b_k(\omega) = (-i\omega)^k,$$

$$(5.14) \quad c_n(\sigma, \omega, c\tau) = \sum_{j=0}^n a_j(\sigma, c\tau) b_{n-j}(\omega) = \sum_{j=0}^n a_j(\sigma, c\tau) (-i\omega)^{n-j}.$$

The recurrence relations (5.8) and (5.9) easily follow from (5.12)–(5.14).

Therefore, for $\Lambda > \sigma$, we obtain

$$\Psi_2(x; \sigma, \omega; \Lambda) = -i \sum_{n=0}^{\infty} c_n(\sigma, \omega, c\tau) \int_{\Lambda}^{\infty} e^{ix\eta} \eta^{-1-2c\tau-n} d\eta.$$

If we set

$$\mathcal{I}(x, z; \Lambda) = \int_{\Lambda}^{\infty} e^{ix\eta} \eta^{-1-z} d\eta,$$

for $z > 0$, then we can write

$$\Psi_2(x; \sigma, \omega; \Lambda) = -i \sum_{n=0}^{\infty} c_n(\sigma, \omega, c\tau) \mathcal{I}(x, 2c\tau + n; \Lambda).$$

Let us now calculate the value of the integral $\mathcal{I}(x, z; \Lambda)$. Put $t = -ix\eta$, so that $\eta = ix^{-1}t$. If $x > 0$, we have

$$\mathcal{I}(x, z; \Lambda) = \int_{-ix\Lambda}^{-i\infty} e^{-t} (ix^{-1}t)^{-1-z} ix^{-1} dt.$$

Therefore we can write

$$(5.15) \quad \mathcal{I}(x, z; \Lambda) = (-ix)^z \int_{-ix\Lambda}^{-i\infty} e^{-t} t^{-1-z} dt = (-ix)^z \Gamma(-z, -ix\Lambda),$$

For $x < 0$ we obtain in a similar way

$$\mathcal{I}(x, z; \Lambda) = (-ix)^z \int_{-ix\Lambda}^{i\infty} e^{-t} t^{-1-z} dt = (-ix)^z \Gamma(-z, -ix\Lambda).$$

Hence in both cases

$$(5.16) \quad \mathcal{I}(x, z; \Lambda) = (-ix)^z \Gamma(-z, -ix\Lambda).$$

□

5.1.1. *Recommendations for the computation of the incomplete gamma function.* The value of the incomplete gamma function $\Gamma(s, y)$ can be calculated very efficiently by noting that the derivative of the lower incomplete gamma function $\gamma(s, y) = \Gamma(s) - \Gamma(s, y)$ is given by $\partial_y \gamma(s, y) = e^{-y} y^{s-1}$, from which the following series expansion can be obtained

$$(5.17) \quad \Gamma(s, y) = \Gamma(s) - e^{-y} y^s \sum_{n=0}^{\infty} \frac{x^n}{s(s+1)\dots(s+n-1)}.$$

This converges very fast when $|y/s| < 1$, which is normally the case, since $s = -2c\tau - n$, $y = -ix\Lambda$, and the method is usually applied when x' is small. If $|y/s| \gg 1$, other techniques may be used, such as Legendre's continued fraction or asymptotic expansions [26]. All the numerical examples in this paper were calculated using the expansion (5.17).

5.1.2. *Error bounds for the remainder of the series expansion (5.10).* The following results are analogous to Propositions 3.3 and 3.4 for the ATM case, and can be used to select the minimum number of terms N to include in the expansion (5.10) for a given error tolerance. The proofs are left to the reader.

Proposition 5.2. *Denote by $\Psi_2^{(N)}(x; \sigma, \omega; \Lambda)$ the approximation to $\Psi_2(x; \sigma, \omega; \Lambda)$ obtained by discarding all terms from $n = N + 1$ onwards in the expansion (5.10). Then the error of approximation of $\Psi_2(x; \sigma, \omega; \Lambda)$ by $\Psi_2^{(N)}(x; \sigma, \omega; \Lambda)$ admits the following bound*

$$(5.18) \quad |\Psi_2^{(N)}(x; \sigma, \omega; \Lambda) - \Psi_2(x; \sigma, \omega; \Lambda)| \leq \psi_N(\Lambda) \frac{\Lambda^{-1-2c\tau-N}}{1+2c\tau+N},$$

where

$$(5.19) \quad \begin{aligned} \psi_N(\Lambda) = & \frac{|a_{N+1}(\sigma, c\tau)|}{\Lambda^N} \frac{\Lambda^{N+1} - |\omega|^{N+1}}{\Lambda - |\omega|} + |\omega|^{N+1} \sum_{n=0}^N |a_n(\sigma, c\tau)| \Lambda^{-n} \\ & + \frac{|\omega|^{N+1} a_{N+1}(\sigma, c\tau)|}{\Lambda^{N+1}} + \sum_{n=0}^{N-1} |c_{n+N+1}(\sigma, \omega, c\tau)| \Lambda^{-n}. \end{aligned}$$

Proposition 5.3. *If $c\tau \leq 1$, the error of approximation of $\Psi_2(x; \sigma, \omega; \Lambda)$ by $\Psi_2^{(N)}(x; \sigma, \omega; \Lambda)$ admits the following bound*

$$(5.20) \quad |\Psi_2^{(N)}(x; \sigma, \omega; \Lambda) - \Psi_2(x; \sigma, \omega; \Lambda)| \leq \tilde{\psi}_N(\Lambda) \frac{\Lambda^{-1-2c\tau-N}}{1+2c\tau+N}.$$

Here

(5.21)

$$\begin{aligned} \tilde{\psi}_N(\Lambda) = & E(N+1) \frac{\sigma^{N+1}}{\Lambda^N} \frac{\Lambda^{N+1} - |\omega|^{N+1}}{\Lambda - |\omega|} + D \left(\frac{N'}{2} \right) |\omega|^{N+1} \frac{1 - \Lambda^{N'-1}}{1 - \Lambda^{-2}} \\ & + E(N+1) \frac{\sigma^{N+1} |\omega|^{N+1} \delta_{N+1}}{\Lambda^N} + \frac{D[(N-1)']}{2(1 - \frac{\sigma^2}{\omega^2})} \left(\frac{\Lambda^N - |\omega|^N}{\Lambda - |\omega|} - \frac{\sigma}{|\omega|} \frac{\Lambda^N - \sigma^N}{\Lambda - \sigma} \right) \Lambda^{1-N}, \end{aligned}$$

$$E(n) = \min[1, c\tau n^{-1}(2 + c\tau(n-2))],$$

$D(n)$ is given by (3.19), $\delta_n = [1 + (-1)^n]/2$, and the prime denotes the smallest even integer, e.g. for integer n we have $n' = n - [1 + (-1)^n]/2$ (of course, the bound still holds if one replaces primed by unprimed quantities).

5.2. **Case $\omega \in [-1, 0]$.** For $\omega \in (-1, 0)$, the same results of Subsection 3.3.1 apply. If, on the other hand, either $\omega = -1$ or $\omega = 0$, then we define

$$\Psi_1(x; \sigma, 0; \Lambda) = -\frac{i}{2} P \int_{-\Lambda}^{\Lambda} e^{ix\eta} \frac{(\sigma^2 + \eta^2)^{-c\tau}}{\eta} d\eta.$$

This can be re-written as

$$(5.22) \quad \Psi_1(x; \sigma, 0; \Lambda) = \frac{\pi i}{2} \mathcal{H}\psi(0),$$

where

$$(5.23) \quad \psi(\eta) = e^{ix\eta} (\sigma^2 + \eta^2)^{-c\tau} \mathbb{1}_{[-\Lambda, \Lambda]}(\eta),$$

and \mathcal{H} denotes the Hilbert transform, defined by

$$\mathcal{H}\psi(\xi) = \frac{1}{\pi} P \int_{-\infty}^{\infty} \frac{\psi(\eta)}{\xi - \eta} d\eta.$$

As noted in [14], the Hilbert transform can be calculated very efficiently by using an approximation based on Whittaker's cardinal series (sinc expansion), which relies on the FFT realization of Toeplitz matrix-vector multiplication. The details of the implementation can be found in Appendix G, and a discussion of the discretization error bound for the calculation of $\Psi_1(x; \sigma, 0; \Lambda)$ using (5.22) can be found in Appendix H.

5.3. **Algorithm.** From the results of the previous sections, we can obtain a procedure to calculate the non-ATM price or delta of a vanilla option, or the non-ATM price of a digital option. We outline the algorithm for vanilla prices. The modifications for the other cases are left to the reader (cf. Appendix F).

1. Fix a value of Λ . As we will see in Subsection 5.5.1, the value of Λ is largely irrelevant to the accuracy of the results. For a fast implementation we recommend taking Λ in the range 4σ - 5σ .

2. Fix the value of the error tolerances ϵ_1 and ϵ_2 , corresponding to the calculations involving $\Psi_1(x'; \sigma, \omega; \Lambda)$ and $\Psi_2(x'; \sigma, \omega; \Lambda)$, respectively (cf. Section 3.4). As in the ATM case, formally $\epsilon = \epsilon_1 + \epsilon_2$, where ϵ is the absolute error tolerance for the option price. However, as we will see in Section 5.5.2, it is usually safe to choose ϵ_1 five or six orders of magnitude larger than ϵ , and ϵ_2 one or two orders of magnitude larger than ϵ . Set $\epsilon'_1 = \epsilon_1 \pi K^{-1} e^{r'\tau + \omega x'}$, and $\epsilon'_2 = \epsilon_2 \pi K^{-1} e^{r'\tau + \omega x'}$.
3. Calculate $\Psi_1(x'; \sigma, \omega; \Lambda)$ and $\Psi_1(x'; \sigma, \omega + 1; \Lambda)$ (cf. definition (5.5)), e.g. by using one of the numerical integration methods discussed in Appendix B, with error tolerance $\epsilon'_1/2$.
4. Calculate $\Psi_2(x'; \sigma, \omega; \Lambda)$ and $\Psi_2(x'; \sigma, \omega + 1; \Lambda)$ (cf. definition (5.6)) by Proposition 5.1, i.e. using the asymptotic expansion (3.11) with N terms, where N can be determined either from Proposition 5.3, if $c\tau \leq 1$, or from Proposition 5.2, if $c\tau > 1$. Use error tolerance $\epsilon'_2/2$.
5. Calculate the price by using (5.1)–(5.6), applying put-call-parity, Proposition 3.5, or the method in Subsection 5.2, if needed.

5.4. Digitals and deltas. The algorithm can easily be modified to price digital options, or to calculate the delta of a vanilla option, with $x' \neq 0$. The details can be found in Appendix F. In each of these cases, we expect the CPU time to be approximately halved compared to the case of the vanilla option price, since one only needs to calculate the value of one improper integral, rather than two.

5.5. Numerical results.

5.5.1. *Choice of Λ .* As in the ATM case, the method has an almost free parameter Λ . In Figure 5 (cf. Figure 1 for the ATMVG case) we plot the relative differences w.r.t. $\Lambda_0 = 4.5\sigma$ and the corresponding CPU times for the calculation of the benchmark FastVG price¹³ of an OTM call option with $x' = -10^{-6}$, under the same VG parameters used in Section 3.5 (as we will see in Section 5.5.3, the performance of the FastVG algorithm is largely independent of the choice of x'). They show that, as in the ATMVG case, the accuracy of the algorithm is independent of Λ , but that larger Λ generally leads to longer calculation times. In what follows, we will take $\Lambda = 4.5\sigma$ for the calculation of the FastVG price, unless stated otherwise¹⁴.

¹³See Subsection 3.5 for the definition of the benchmark.

¹⁴As in Subsection 3.5, all the plots in this section were generated using adaptive Gauss-Lobatto quadrature for the calculation of $\Psi_1(x'; \sigma, \omega; \Lambda)$. As shown in Appendix C.2, other methods are much faster, but adaptive Gauss-Lobatto quadrature can cope much more easily with large ranges of integration. Therefore, the plots of CPU times in the remainder of this section should be used only to judge the *relative* performance of the algorithm for different choices of truncation parameter Λ and error tolerances ϵ_1 and ϵ_2 . The CPU times corresponding to other integration methods can be found in Appendix C.2.

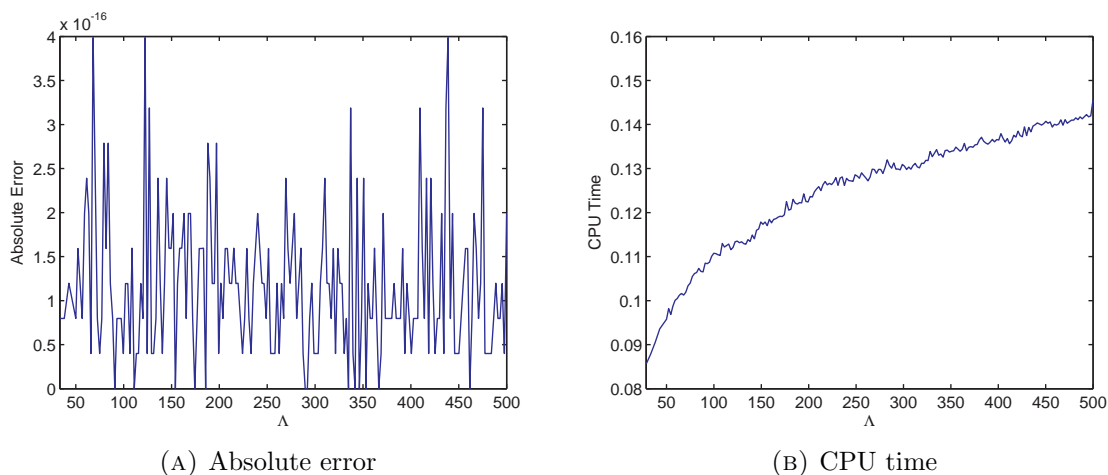


FIGURE 5. $|V_{\text{bm}}(\Lambda) - V_{\text{bm}}(\Lambda_0)|$, and corresponding CPU times, for varying Λ and $\Lambda_0 = 4.5\sigma$, for an OTM vanilla call option with $x' = -10^{-6}$, $K = 1$, $\lambda_- = -11$, $\lambda_+ = 8$, $m_2 = 0.16$, $r = 0.03$, $q = 0$, $\tau = 0.004$, for $\epsilon_1 = \epsilon_2 = 10^{-16}$.

5.5.2. *Choice of ϵ_1 and ϵ_2 .* Figures 6–7 show the effect of varying the error tolerances ϵ_1 and ϵ_2 used in the calculation of $\Psi_1(x'; \sigma, \omega; \Lambda)$ and $\Psi_2(x'; \sigma, \omega; \Lambda)$, respectively, on price accuracy and CPU time (cf. Figures 2–3 for the ATMVG case). As in Subsection 5.5.1, we take an OTM call option with $x' = -10^{-6}$. We note that, similarly to the ATMVG case

1. The price does not change between $\epsilon_1 = 10^{-16}$ and $\epsilon_1 \simeq 2 \cdot 10^{-11}$, and the absolute difference w.r.t. $\epsilon_1 = 10^{-16}$ is still well below floating point accuracy up to $\epsilon_1 \simeq 2.5 \cdot 10^{-10}$.
2. The CPU time falls almost monotonically for decreasing ϵ_1 .
3. The absolute error increases approximately monotonically for increasing ϵ_2 , staying about two orders of magnitude below ϵ_2 , and the CPU time is largely independent of ϵ_2 .

We conclude that, even if accuracy of the order of floating point precision is needed, one might as well take ϵ_1 in the range 10^{-11} – 10^{-7} and ϵ_2 in the range 10^{-15} – 10^{-14} .

5.5.3. *Comparison of CPU times.* For comparisons of the performances obtained using different methods, see Appendix C.2. In Figure 8 we plot the CPU times taken to price an OTM call option with the FastVG method, for a wide range of x' from $x' = -0.01$ to $x' = -10^{-10}$. It can be seen that the CPU time is almost independent of x' . This indicates that the conclusions from the previous examples, for $x' = -10^{-6}$, will also hold for other values of x' .

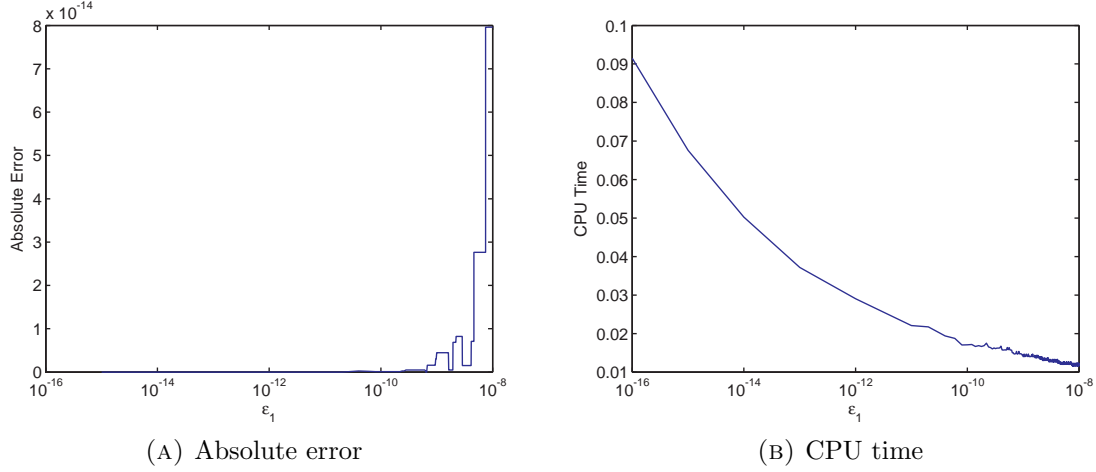


FIGURE 6. Absolute errors in the FastVG price, and corresponding CPU times, for varying error tolerance level ϵ_1 w.r.t. $\epsilon = 10^{-16}$, for an OTM vanilla call option with $x' = -10^{-6}$, $K = 1$, $\lambda_- = -11$, $\lambda_+ = 8$, $m_2 = 0.16$, $r = 0.03$, $q = 0$, $\tau = 0.004$, for $\Lambda = 4.5\sigma$.

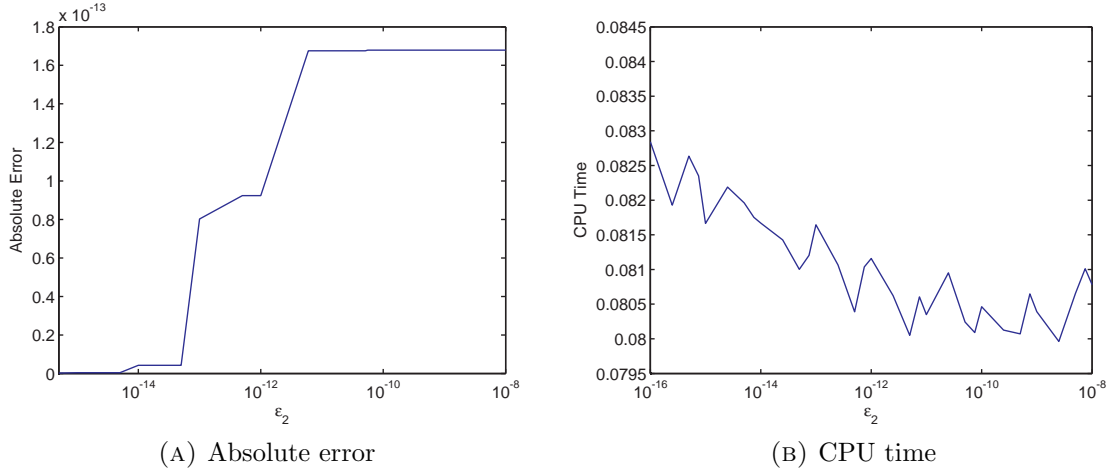


FIGURE 7. Absolute errors in the ATMVG price, and corresponding CPU times, for varying error tolerance level ϵ_2 w.r.t. $\epsilon = 10^{-16}$, for an OTM vanilla call option with $x' = -10^{-6}$, $K = 1$, $\lambda_- = -11$, $\lambda_+ = 8$, $m_2 = 0.16$, $r = 0.03$, $q = 0$, $\tau = 0.004$, for $\Lambda = 4.5\sigma$.

In Figure 9 we show the CPU times taken by parabolic iFT with $\alpha = 3$ (solid line) versus those of FastVG with Clenshaw-Curtis integration, $\epsilon_1 = 10^{-7}$, $\epsilon_2 = 10^{-15}$ (dashed line), for better than 1% accuracy (left panel) and better than 0.1% accuracy

(right panel), with the following parameter set, which was fitted to the volatility surface of AUDUSD options as of 12 March 2010: $\lambda_- = -31.6586$, $\lambda_+ = 14.9279$, $m_2 = 0.0232$. In both cases, we take a vanilla option with $K = 1$, $r = 0.03$, $q = 0$, and maturity $\tau = 0.004$ (approximately one day). Our results show that, for vanilla option prices, FastVG is faster than parabolic iFT for short maturities and options close to ATM, or for small error tolerances. As noted in the introduction, such short distances from the ATM point $x' = 0$ may naturally occur during a model calibration, when the optimization hits a region of parameter space with near-zero drift.

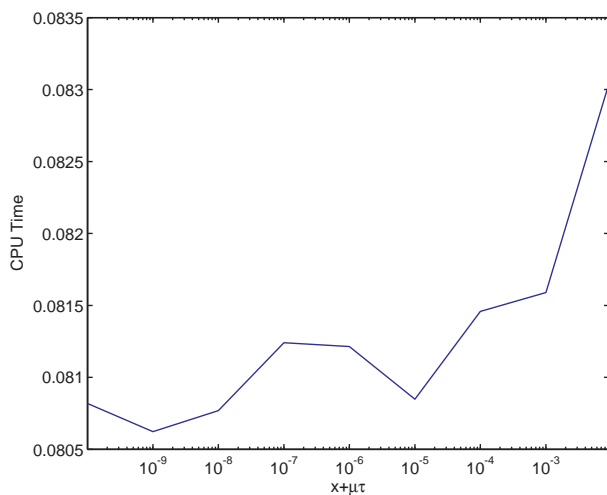


FIGURE 8. CPU times taken for the calculation of the FastVG price for an OTM vanilla call option with $K = 1$, $\lambda_- = -11$, $\lambda_+ = 8$, $m_2 = 0.16$, $r = 0.03$, $q = 0$, $\tau = 0.004$, for $\Lambda = 4.5\sigma$, for varying $x' = x + \mu\tau$, using Gauss-Lobatto quadrature.

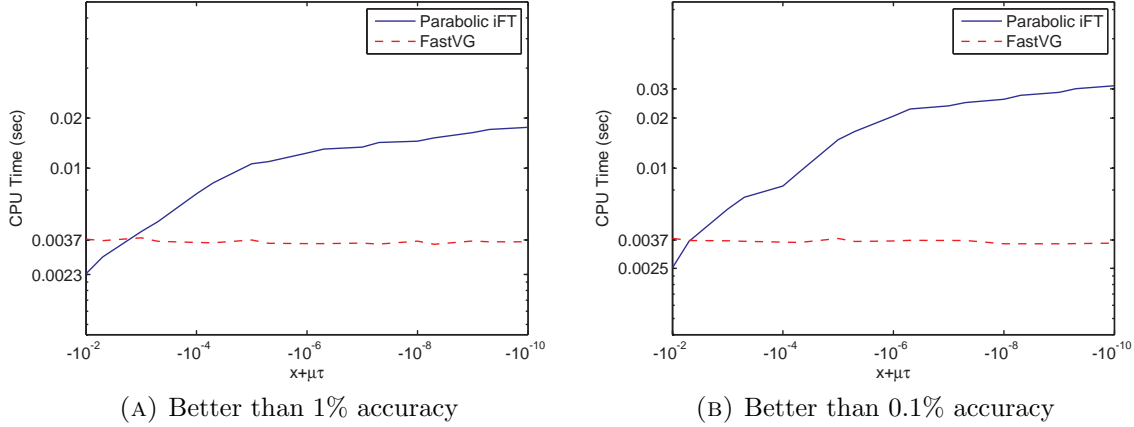


FIGURE 9. Logarithmic plots of CPU times with parabolic iFT (with $\alpha = 3$) and FastVG for the price of a vanilla call option with $K = 1$, $\lambda_- = -31.6586$, $\lambda_+ = 14.9279$, $m_2 = 0.0232$, $r = 0.03$, $q = 0$, $\tau = 0.004$, for varying $x' = x + \mu\tau$, with different accuracy levels. The VG parameters were calibrated to the volatility surface of AUDUSD options as of 12 March 2010.

5.5.4. *CPU times for digitals.* In Figure 10 we show the CPU times taken by parabolic iFT with $\alpha = 3$ (solid line) versus those of FastVG with Clenshaw-Curtis integration, $\epsilon_1 = 10^{-7}$, $\epsilon_2 = 10^{-15}$ (dashed line), with better than 1% accuracy (left panel) and better than 0.1% accuracy (right panel), for maturity $\tau = 0.004$. In both cases, we take a digital call option with $K = 1$, $r = 0.03$, $q = 0$, and the same process parameters used in Subsection 5.5.3, namely $\lambda_- = -31.6586$, $\lambda_+ = 14.9279$, $m_2 = 0.0232$. In general, FastVG outperforms parabolic iFT for maturities up to about two weeks, except for deep OTM options.

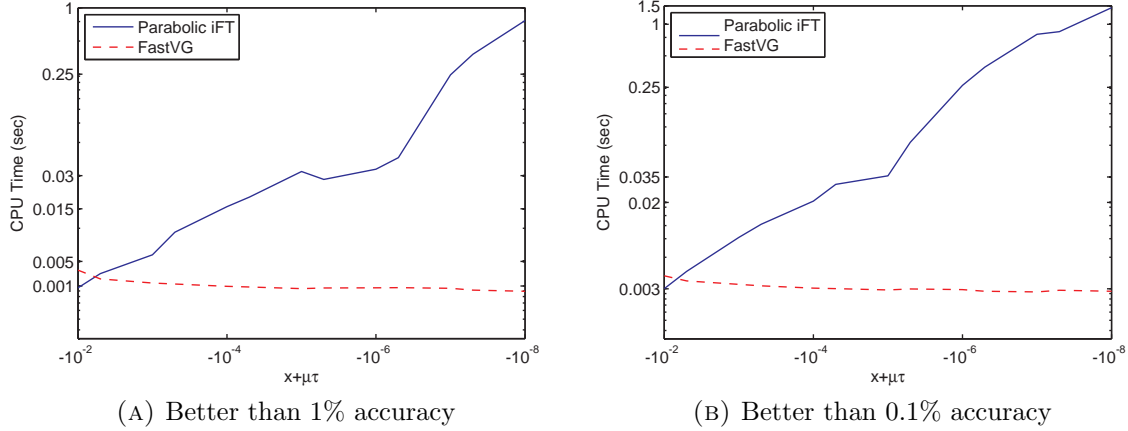


FIGURE 10. Logarithmic plots of CPU times with parabolic iFT (with $\alpha = 3$) and FastVG for the price of a digital call option with $K = 1$, $\lambda_- = -31.6586$, $\lambda_+ = 14.9279$, $m_2 = 0.0232$, $r = 0.03$, $q = 0$, $\tau = 1/26$, for varying $x' = x + \mu\tau$, with different accuracy levels.

5.6. Vanilla gamma, digital delta and VG PDF. A simple modification of the FastVG algorithm can be used to calculate the gamma of a non-ATM vanilla option, the delta of a non-ATM digital option, or the probability density function $p_\tau(x)$ of the VG process X_τ for $x \neq 0$ (cf. Appendix F) by re-using the program for the calculation of $\mathcal{I}(x, z; \Lambda)$ in (5.16).

The gamma of a vanilla option is given by equation (F.7), which can be re-written as

$$(5.24) \quad \text{Gamma}(\tau, x) = \frac{e^{-r'\tau - \omega x' - 2x}}{2\pi K} \Omega(x'; \sigma),$$

where

$$(5.25) \quad \Omega(x; \sigma) = \int_{-\infty}^{\infty} e^{ix\eta} (\sigma^2 + \eta^2)^{-c\tau} d\eta.$$

This can be expressed in terms of the VG PDF $p_\tau(x)$ as follows

$$(5.26) \quad \text{Gamma}(\tau, x) = K e^{-r\tau - 2x} p_\tau(x).$$

It is well-known that $p_\tau(x)$ can be expressed in closed form¹⁵. We have the following result.

¹⁵The expression for the PDF of the VG distribution was obtained by Pearson in 1929 [22] and introduced to finance by Madan, Carr and Chang in 1998 [20].

Proposition 5.4. *The gamma of a vanilla option under VG, for $x \neq 0$, is given by*

$$(5.27) \quad \text{Gamma}(\tau, x) = \frac{e^{-r'\tau - \omega x' - 2x}}{\pi} \Omega(x'; \sigma) d\eta,$$

where

$$(5.28) \quad \Omega(x; \sigma) = \frac{2^{\frac{3}{2}-c\tau} \sqrt{\pi} \sigma^{\frac{1}{2}-c\tau} |x|^{-\frac{1}{2}+c\tau} K_{\frac{1}{2}-c\tau}(\sigma|x|)}{\Gamma(c\tau)},$$

where $K_\nu(x)$ denotes the modified Bessel function of the second kind.

Proof. Follows from the known expression for the VG PDF. Alternatively, one can write (5.25) as

$$(5.29) \quad \Omega(x; \sigma) = 2 \operatorname{Re} \int_0^\infty e^{ix\eta} (\sigma^2 + \eta^2)^{-c\tau} d\eta = 2 \int_0^\infty \cos(x\eta) (\sigma^2 + \eta^2)^{-c\tau} d\eta.$$

The result then follows from the integral representation of $K_\nu(x)$ [27, §6.16]

$$K_\nu(xz) = \frac{\Gamma\left(\nu + \frac{1}{2}\right) \cdot (2z)^\nu}{x^\nu \Gamma\left(\frac{1}{2}\right)} \int_0^\infty \frac{\cos(xu)}{(z^2 + u^2)^{\nu+\frac{1}{2}}} du,$$

for $\operatorname{Re} \nu \geq -1/2$, $x > 0$, $|\arg z| < \pi/2$. \square

An alternative approach, which only relies on the computation of the incomplete gamma function, rather than the modified Bessel function of the second kind, is to calculate $\Omega(x; \sigma)$ by a similar method to that of Section 5. If we write $\Omega(x; \sigma) = \Omega_1(x; \sigma; \Lambda) + \Omega_2(x; \sigma; \Lambda)$, where

$$\begin{aligned} \Omega_1(x; \sigma; \Lambda) &= 2 \operatorname{Re} \int_0^\Lambda e^{ix\eta} (\sigma^2 + \eta^2)^{-c\tau} d\eta, \\ \Omega_2(x; \sigma; \Lambda) &= 2 \operatorname{Re} \int_\Lambda^\infty e^{ix\eta} (\sigma^2 + \eta^2)^{-c\tau} d\eta, \end{aligned}$$

and $\Lambda > \sigma$, then the following result can be obtained.

Proposition 5.5. *The value of $\Omega_2(x; \sigma; \Lambda)$, for $\Lambda > \sigma$ and $\sigma > 0$, $x \neq 0$, can be calculated using the series expansion*

$$(5.30) \quad \Omega_2(x; \sigma; \Lambda) = \eta^{-2c\tau} \sum_{n=0}^{\infty} \alpha_n(\sigma, c\tau) \mathcal{I}(x, 2c\tau + 2n - 1; \Lambda),$$

where $\alpha_n(\sigma, c\tau)$ is given by (5.11) and $\mathcal{I}(\cdot, \cdot; \cdot)$ by (5.15).

Proof. Follows from the proofs of Propositions 3.1 and 5.1. \square

This result can be used to derive an algorithm for the calculation of $\Omega_2(x; \sigma; \Lambda)$ similar to the one used for $\Psi_2(x; \sigma, \omega; \Lambda)$ in Section 5. As in the cases of $\Phi_1(\sigma, \omega; \Lambda)$

and $\Psi_1(x; \sigma, \omega; \Lambda)$, the value of $\Omega_1(x; \sigma; \Lambda)$ can be calculated using one of several numerical adaptive procedures (cf. Appendix B). We state without proof the following error bounds, analogous to the ones for $\Phi_2^{(N)}(\sigma, \omega; \Lambda)$ and $\Psi_2^{(N)}(x; \sigma, \omega; \Lambda)$.

Proposition 5.6. *Denote by $\Omega_2^{(N)}(x; \sigma; \Lambda)$ the approximation to $\Omega_2(x; \sigma; \Lambda)$ obtained by discarding all terms from $n = N + 1$ onwards in the expansion (5.30). Then*

$$(5.31) \quad |\Omega_2^{(N)}(x; \sigma; \Lambda) - \Omega_2(x; \sigma; \Lambda)| \leq 2|\alpha_{N+1}(\sigma, c\tau)|\mathcal{I}(x, 1 + 2c\tau + 2N; \Lambda)$$

$$(5.32) \quad \leq 2|\alpha_{N+1}(\sigma, c\tau)|\frac{\Lambda^{-1-2c\tau-2N}}{1 + 2c\tau + 2N}.$$

If $c\tau \leq 1$, then the previous bound can be replaced by

$$(5.33) \quad |\Omega_2^{(N)}(x; \sigma; \Lambda) - \Omega_2(x; \sigma; \Lambda)| \leq 2C(N + 1)\sigma^{2N+2}\mathcal{I}(x, 1 + 2c\tau + 2N; \Lambda)$$

$$(5.34) \quad \leq 2C(N + 1)\sigma^{2N+2}\frac{\Lambda^{-1-2c\tau-2N}}{1 + 2c\tau + 2N},$$

where $C(n)$ is given by (3.18).

Remark 5.7. We note that, if a sufficiently large Λ is chosen, convergence of the series (5.30) is normally very fast, and only 1-2 terms are needed for extremely high accuracy.

5.6.1. *Algorithm.* The straightforward modification of the algorithm in Section 5.3 is left to the reader. We only note that, unlike the cases of Sections 3 and 5, the procedure does not depend on the value of ω .

5.6.2. *Numerical results.* We omit the examples on the choice of Λ , error tolerances, and the comparisons between different integration methods. We note however that, due to the slower convergence of the integrand, it is better to take a slightly larger Λ than in the previous cases, usually around 10σ . In Figure 11 we show the CPU times taken by parabolic iFT with $\alpha = 3$ (solid line) versus those of FastVG with Clenshaw-Curtis integration, $\epsilon_1 = 10^{-6}$, $\epsilon_2 = 10^{-15}$ (dashed line), for the calculation of gamma with better than 1% accuracy (left panel) and better than 0.1% accuracy (right panel), for time to maturity $\tau = 1/12$, and the same parameter set used in Subsections 5.5.3 and 5.5.4 for the prices of vanilla and digital options, respectively. The values of the error tolerances ϵ_1 and ϵ_2 were chosen to ensure that the absolute error of the FastVG gamma w.r.t. the one calculated with the closed form solution from Proposition 5.4 was of the order of 1-10 times floating point accuracy. These pictures show that, even for maturities as long as one month, FastVG is faster and more accurate than parabolic iFT for the calculation of gamma, even very far away from ATM. The same conclusions hold for the delta of a digital option. Figure 12 shows the CPU times taken by FastVG compared with those taken by direct calculation of the function $K_\nu(x)$ using two series expansions for the modified Bessel function of the first kind [1]. In general, the expansion (5.17) for the incomplete

gamma function converges faster than the ones used for the calculation of $K_\nu(x)$; however, FastVG also requires computing the integral $\Omega_1(x; \sigma; \Lambda)$. On the whole, the overall CPU time taken by both methods is approximately the same. The relative differences between the values of gamma calculated by the two methods are of the order of 10^{-16} – 10^{-15} .

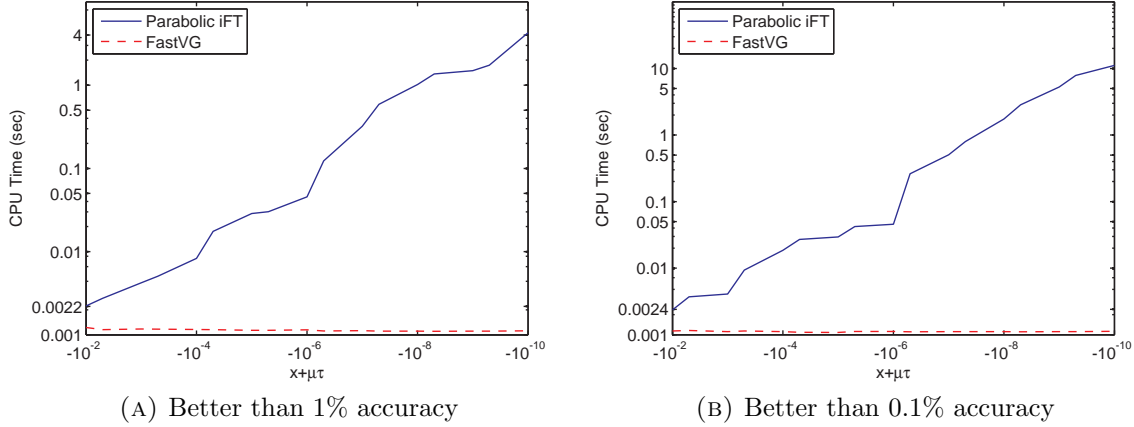


FIGURE 11. Logarithmic plots of CPU times with parabolic iFT (with $\alpha = 3$) and FastVG for the gamma of a vanilla call option with $K = 1$, $\lambda_- = -31.6586$, $\lambda_+ = 14.9279$, $m_2 = 0.0232$, $r = 0.03$, $q = 0$, $\tau = 1/12$, for varying $x' = x + \mu\tau$, with different accuracy levels.

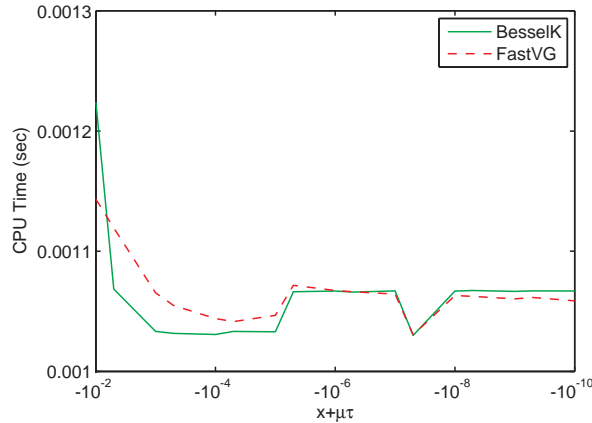


FIGURE 12. Comparison of CPU times with FastVG and direct calculation of the modified Bessel function (BesselK) for the gamma of a vanilla call option with $K = 1$, $\lambda_- = -31.6586$, $\lambda_+ = 14.9279$, $m_2 = 0.0232$, $r = 0.03$, $q = 0$, $\tau = 1/12$, for varying $x' = x + \mu\tau$.

6. CONCLUSIONS

We introduced a new method (ATMVG) to calculate the price of ATM European options under the VG model, and showed that it is much faster and more accurate than all versions of iFT, including the ones recently introduced in [5].

Using the ATMVG price as a benchmark, we compared the ATM prices of short maturity vanilla options calculated with flat, parabolic, and hyperbolic iFT. As expected from the analysis in [5], we found that parabolic iFT is much more efficient than the other two methods, and its errors w.r.t. ATMVG can be made as small as 10^{-14} or 10^{-15} .

We looked at the problem of interpolation near ATM, which is of particular relevance for model fitting, especially when flat iFT or one of its common variants is used. Comparing several different methods, we showed that linear interpolation, normally used in model calibration, can give large errors near ATM. Somewhat better accuracy can be obtained by using polynomial interpolation and taking the option deltas into account as well. For digitals, on the other hand, interpolation near ATM can produce very inaccurate results, even taking account of both prices and deltas at several points.

We introduced the FastVG method to calculate the price or delta of a non-ATM European option, or the price of a digital option, under the VG model. Despite being slightly slower than ATMVG, and having to rely on numerical calculation of the incomplete gamma function, this procedure is both very accurate and much faster for short maturity, near ATM options than parabolic iFT, particularly for the digital price and the vanilla delta. In these cases, FastVG can replace iFT for applications (such as model fitting or risk management) in which speed is an important concern. The same can be done with the ATMVG method for ATM options. We also showed how a variation of the FastVG algorithm can be used to calculate the gamma of a non-ATM vanilla option, the delta of a non-ATM digital option, or the PDF of the VG process (which is particularly useful for historical calibration). This provides an alternative to the well-known procedure which relies on the computation of the modified Bessel function of the second kind.

Finally, we note that all numerical examples included in this paper were calculated with MATLAB on a home computer. The performance of the algorithms when implemented in optimized C++ code will typically be around 10-100 times faster.

APPENDIX A. TRUNCATION ERROR BOUND FOR THE VG OPTION PRICE

We truncate the integral on the RHS of (3.1) as follows

$$(A.1) \quad V(\tau, x; \Lambda) = -\frac{e^{-r'\tau - \omega x'}}{2\pi} \int_{-\Lambda}^{\Lambda} e^{ix'\eta} \frac{(\sigma^2 + \eta^2)^{-c\tau}}{(\eta + i\omega)(\eta + i\omega + i)} d\eta.$$

We refer to $V(\tau, x; \Lambda)$ as the truncated price and to $\text{Tr.Err.}(V; \tau, x; \Lambda) = |V(\tau, x; \Lambda) - V(\tau, x)|$ as the (absolute) truncation error. A useful bound for the truncation error is given by the following result.

Proposition A.1. $\text{Tr.Err.}(V; \tau, x; \Lambda)$ admits the following bound

$$(A.2) \quad \text{Tr.Err.}(V; \tau, x; \Lambda) \leq C(\Lambda) \frac{e^{-r'\tau - \omega x'}}{\pi(1 + 2c\tau)} \Lambda^{-1-c\tau},$$

where $C(\Lambda) = {}_2F_1\left(\frac{1}{2} + c\tau, 1 + c\tau, \frac{3}{2} + c\tau, -\frac{\omega^2}{\Lambda^2}\right)$, and ${}_2F_1(\cdot, \cdot, \cdot, \cdot)$ is the hypergeometric function.

Proof. Putting $\alpha = \pi^{-1}e^{-r'\tau - \omega x'}$, we have

$$\begin{aligned} \text{Tr.Err.}(\tau, x; \Lambda) &= \left| \alpha \operatorname{Re} \int_{\Lambda}^{+\infty} \frac{e^{ix'\eta}(\sigma^2 + \eta^2)^{-c\tau}}{(\eta + i\omega)(\eta + i\omega + i)} d\eta \right| \leq \alpha \int_{\Lambda}^{+\infty} \left| \frac{e^{ix'\eta}(\sigma^2 + \eta^2)^{-c\tau}}{(\eta + i\omega)(\eta + i\omega + i)} \right| d\eta \\ &\leq \alpha \int_{\Lambda}^{+\infty} \frac{(\sigma^2 + \eta^2)^{-c\tau}}{|\eta + i\omega|^2} d\eta \leq \alpha \int_{\Lambda}^{+\infty} (\omega^2 + \eta^2)^{-1-c\tau} d\eta \\ &= \frac{\alpha \Lambda^{-1-c\tau}}{1 + 2c\tau} {}_2F_1\left(\frac{1}{2} + c\tau, 1 + c\tau, \frac{3}{2} + c\tau, -\frac{\omega^2}{\Lambda^2}\right). \end{aligned}$$

The last equality can be proved by recalling that, according to the Gauss contiguous relations for the hypergeometric function, one has

$$z \frac{d}{dz} {}_2F_1(a, b, c, z) = (c - 1)({}_2F_1(a, b, c - 1, z) - {}_2F_1(a, b, c, z)).$$

Therefore, if γ is any complex number, we have

$$\begin{aligned} \frac{d}{d\eta} {}_2F_1\left(\frac{1}{2}, \gamma, \frac{3}{2}, -\eta^2\right) &= \eta^{-1} \left({}_2F_1\left(\frac{1}{2}, \gamma, \frac{1}{2}, -\eta^2\right) - {}_2F_1\left(\frac{1}{2}, \gamma, \frac{3}{2}, -\eta^2\right) \right) \\ &= \eta^{-1} \left((1 + \eta^2)^{-\gamma} - {}_2F_1\left(\frac{1}{2}, \gamma, \frac{3}{2}, -\eta^2\right) \right). \end{aligned}$$

This gives us

$$(A.3) \quad \frac{d}{d\eta} \left[\eta {}_2F_1\left(\frac{1}{2}, \gamma, \frac{3}{2}, -\eta^2\right) \right] = (1 + \eta^2)^{-\gamma}.$$

□

Remark A.2. Since $C(\Lambda)$ tends to 1 very rapidly for $\Lambda \rightarrow \infty$, in a practical application one can safely set $C(\Lambda) = 1$ for large Λ .

Remark A.3. This result is similar to Proposition 2.5 in [5], except that the value of the asymptotic constant is now given explicitly.

APPENDIX B. ADAPTIVE INTEGRATION METHODS

In a practical implementation, one would usually calculate an integral such as $\Phi_1(\sigma, \omega; \Lambda)$ by an adaptive quadrature scheme, i.e. one in which the integration grid is successively refined until certain convergence criteria are met, with the aim that the absolute error should lie under a tolerance level ϵ [24, §4.7]. In our numerical examples, we included results calculated with the “adaptive Simpson’s rule” developed in [15], the adaptive Gauss-Lobatto rule of *op. cit.*, and the adaptive Clenshaw-Curtis quadrature, implemented according to the algorithm given in [23].

APPENDIX C. COMPARISON OF DIFFERENT QUADRATURE METHODS

C.1. ATMVG case. We compare the ATMVG price calculated with $\Lambda = 1.5\sigma$, $\epsilon_1 = 10^{-12}$, $\epsilon_2 = 10^{-15}$, using each of the methods in Appendix B, namely: adaptive Simpson’s rule, Clenshaw-Curtis integration, the adaptive Gauss-Lobatto rule (used in previous examples), and MATLAB’s vectorized Gauss-Kronrod quadrature, available as the function `quadgk` and implemented according to the algorithm given in [25]. In Table 1 we list the absolute and relative differences of the ATMVG prices obtained with each of these methods w.r.t. the benchmark price calculated with the adaptive Gauss-Lobatto rule, and the CPU times taken by each. In addition, we list the corresponding results obtained with IAC and parabolic iFT¹⁶. The absolute differences are very small indeed, ranging from $4.58 \cdot 10^{-15}$ for Gauss-Lobatto to $3.98 \cdot 10^{-16}$ for Clenshaw-Curtis and Gauss-Kronrod quadratures. CPU times range from 37.76 ms for Adaptive Simpson integration to 5.08 ms for Clenshaw-Curtis, and 1.84 ms for Gauss-Kronrod quadrature.

Finally, we note that the CPU time taken by the calculation of $\Phi_2(\sigma, \omega; \Lambda)$ in the above example is only 0.56 ms, which shows that most of the time is taken up by the calculation of $\Phi_1(\sigma, \omega; \Lambda)$.

TABLE 1. Differences w.r.t. the benchmark and CPU times for ATMVG calculation of the price of an ATM call option with $K = 1$, $\lambda_- = -11$, $\lambda_+ = 8$, $m_2 = 0.16$, $r = 0.03$, $q = 0$, $\tau = 0.004$.

Method	Abs. Diff.	Rel. Diff.	CPU Time (sec.)
Adapt. Simpson	$3.43 \cdot 10^{-15}$	$1.40 \cdot 10^{-12}$	0.03776
Clenshaw Curtis	$3.98 \cdot 10^{-16}$	$1.62 \cdot 10^{-13}$	0.00486
Gauss-Kronrod	$3.98 \cdot 10^{-16}$	$1.62 \cdot 10^{-13}$	0.00148
Gauss-Lobatto	$4.58 \cdot 10^{-15}$	$1.87 \cdot 10^{-12}$	0.01373
parabolic iFT	$3.17 \cdot 10^{-14}$	$1.29 \cdot 10^{-11}$	12.223

¹⁶Run with $\zeta = 0.125$, $\Lambda = 180000$, $\alpha = 3$. See Appendix D.2.

C.2. FastVG case. We compare the FastVG price calculated with $\Lambda = 4.5\sigma$, $\epsilon_1 = 10^{-7}$, $\epsilon_2 = 10^{-15}$, using different quadrature methods for the calculation of $\Psi_1(x'; \sigma, \omega; \Lambda)$. In addition to the adaptive Gauss-Lobatto rule used in previous examples, we include the other adaptive schemes mentioned in Appendix B: the adaptive Simpson's rule, Clenshaw-Curtis integration, and MATLAB's `quadgk` function (vectorized Gauss-Kronrod quadrature). We also include parabolic iFT (with $\alpha = 3$) and IAC in the comparison. The results, for an OTM vanilla call option with $x' = -10^{-6}$, are shown in Table 2. It can be seen that absolute and relative differences w.r.t. the benchmark (calculated with the adaptive Gauss-Lobatto rule) are very close to floating point accuracy for all integration methods, apart from the adaptive Simpson's rule. The price calculated with parabolic iFT is also extremely close to this benchmark, with an absolute difference of about $1.53 \cdot 10^{-16}$, while the absolute error of the IAC price is much larger ($1.73 \cdot 10^{-10}$). The most efficient FastVG implementations are the ones using Clenshaw-Curtis and Gauss-Kronrod integration, which both take 6.94 ms, whereas parabolic iFT takes 0.28 seconds of CPU time to obtain the same accuracy.

TABLE 2. Differences w.r.t. the benchmark and CPU times for FastVG calculation of the price of an OTM call option with $x' = -10^{-6}$, $K = 1$, $\lambda_- = -11$, $\lambda_+ = 8$, $m_2 = 0.16$, $r = 0.03$, $q = 0$, $\tau = 0.004$.

Method	Abs. Diff.	Rel. Diff.	CPU Time (sec.)
Adapt. Simpson	$6.94 \cdot 10^{-11}$	$2.83 \cdot 10^{-08}$	0.00674
Clenshaw Curtis	$4.38 \cdot 10^{-16}$	$1.79 \cdot 10^{-13}$	0.00694
Gauss-Kronrod	$1.99 \cdot 10^{-16}$	$8.14 \cdot 10^{-14}$	0.00694
Gauss-Lobatto	$7.57 \cdot 10^{-16}$	$3.36 \cdot 10^{-12}$	0.00742
Parabolic iFT	$1.53 \cdot 10^{-16}$	$6.80 \cdot 10^{-13}$	0.28169
IAC	$1.73 \cdot 10^{-10}$	$7.69 \cdot 10^{-7}$	0.00140

APPENDIX D. COMPARISON WITH iFT

D.1. Flat iFT. Flat iFT consists in the computation of a truncated and discretized version of (2.9) over a uniform grid. It is clear from the results of Subsection 3.5 that flat iFT is extremely inefficient for ATM options compared to ATMVG. It is also clear from [17, 5] that flat iFT and its common variants are far less efficient than the new versions introduced in [5], especially parabolic iFT. We only include it in this comparison since it is widely used by practitioners.

Throughout this appendix, when comparing our results with those obtained by iFT, unless stated otherwise, we will use a grid $x_j = -\mu\tau - j\Delta$, $j = 0, \dots, M$ for the latter, with $\Delta = 0.02$, $M = 12$, and $x_M = -\mu\tau$. We fix our initial error tolerance at $\epsilon = 10^{-7}$ and, using (A.2), choose the smallest $\Lambda = \Lambda_0$ such that the truncation error is smaller than $\epsilon/2$. Then, using this Λ_0 , we take an initial mesh $\zeta \gg 1$ in the dual space, calculate the option price, and progressively decrease ζ by a constant factor between 1 and 2, until the difference between two successive evaluations lies below

$\epsilon/2$. The justification for this procedure lies in the fact that, as shown in [5], the error in the discretized version of (2.9), without truncation, decays as $C \exp[-2\pi d(\omega)/\zeta]$, where C is a constant and $d(\omega)$ is the minimum distance between the line $\text{Im}(\xi) = \omega$ and the boundary of the strip of analyticity of the integrand in (2.9). Therefore, by decreasing ζ in a loop by a factor of 1.5 each time, say, we can expect to reach very quickly a point where the discretization error is negligible. Figure 13 shows a logarithmic plot of the absolute pricing error w.r.t. the ATMVG price (solid line) as a function of ζ , when Λ is held fixed at a value chosen to ensure that the truncation error is smaller than $5 \cdot 10^{-10}$ ($\Lambda \simeq 2.441 \cdot 10^8$). The solid line in Figure 13 is therefore a good approximation to the discretization error. The dashed line is just $C \exp[-2\pi d(\omega)/\zeta]$, where C is chosen so that the theoretical estimate coincides with the actual error for $\zeta = 2$. The two curves closely follow each other as the mesh is reduced from $\zeta = 2$ down to about $\zeta = 0.6$ (for lower values of ζ the curves diverge, since the theoretical discretization error for such values of ζ is smaller than residual errors, mostly due to truncation).

Figure 14 shows the truncation error and CPU time as functions of Λ . We note that very large grids are needed for good accuracies, e.g. for a relative pricing error smaller than 1% one needs over 10,000 points, and 60 ms of CPU time. For relative errors smaller than 0.01%, over 120,000 points are needed, and 0.47 seconds of CPU time.

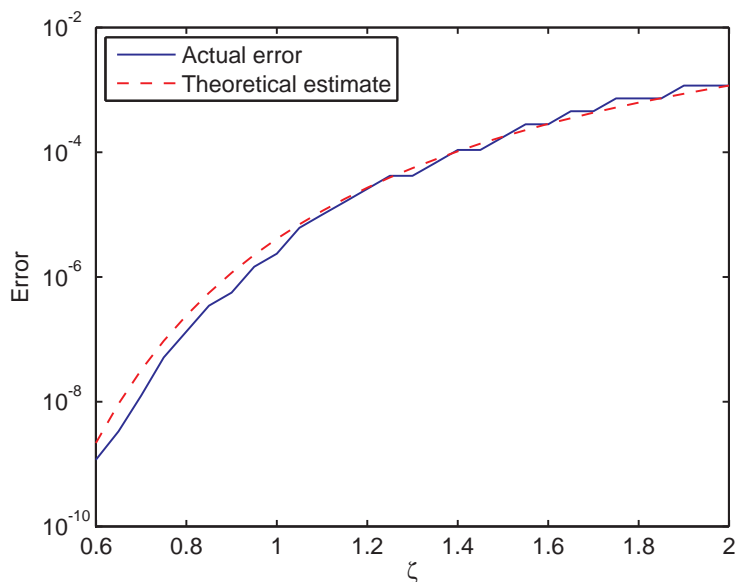


FIGURE 13. Discretization error under flat iFT for an ATM vanilla call option with $K = 1$, $\lambda_- = -11$, $\lambda_+ = 8$, $m_2 = 0.16$, $r = 0.03$, $q = 0$, $\tau = 0.004$, for varying ζ .

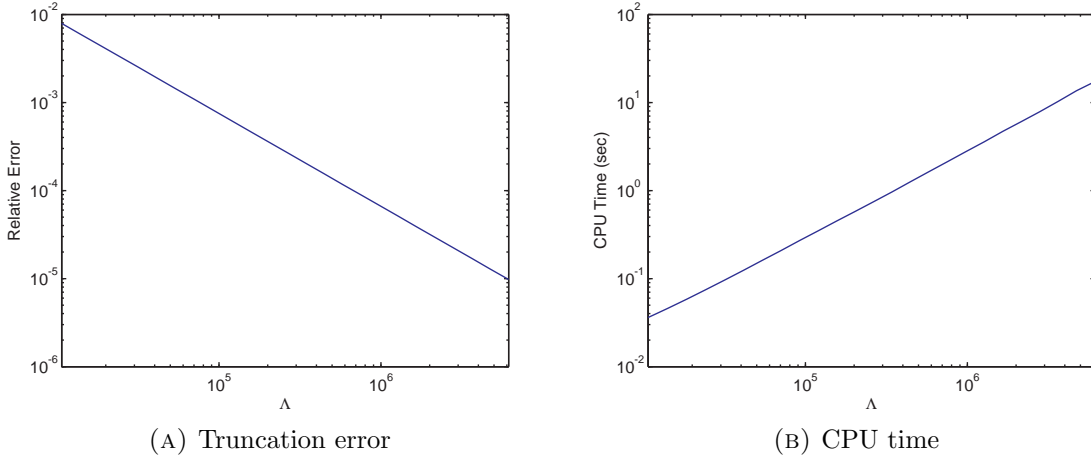


FIGURE 14. Truncation error (logarithmic plot) and CPU time (logarithmic plot) under flat iFT for an ATM vanilla call option with $K = 1$, $\lambda_- = -11$, $\lambda_+ = 8$, $m_2 = 0.16$, $r = 0.03$, $q = 0$, $\tau = 0.004$, for varying Λ .

D.2. Parabolic iFT. This method, introduced in [5], relies on deforming the contour of integration in (2.9) under a conformal map with parameter $\alpha \in (1, 4)$. Details of the method and its implementation can be found in *op. cit.* Theoretical bounds for the ATM truncation error (available on request) can be derived, and the same procedure as for flat iFT can be followed to separate the discretization and truncation errors. Figure 15 shows the absolute truncation error and the CPU time for the price of an ATM vanilla call option with parabolic iFT with $\alpha = 3$. The difference with flat iFT is impressive: parabolic iFT is several orders of magnitude faster and more accurate, needing only 880 points, and 4 ms of CPU time, for relative errors smaller than 1%, and 9535 points, and 34.6 ms of CPU time, for relative errors smaller than 0.01%. Increasing Λ further, one can achieve absolute and relative errors of the order of 10^{-10} to 10^{-9} . Of course, this gain in accuracy for increasing Λ comes at the cost of additional CPU time: already for errors of the order of 10^{-9} to 10^{-8} , the calculation can take 0.10–0.60 seconds, whereas, as we saw in Section 3.5, ATMVG is much faster.

D.3. Hyperbolic iFT. This method, introduced in [5], relies on deforming the contour of integration in the pricing formula by a different conformal map, dependent on a parameter $\alpha > 0$. Again, a theoretical bound for the ATM truncation error under VG (available on request) can be derived, and the same procedure can be used as in the previous sections. The results, for $\alpha = 3.8$, are shown in Figure 16. In general, hyperbolic iFT performs less well than parabolic iFT for ATM options under VG. However, it is still much faster and more accurate than flat iFT.

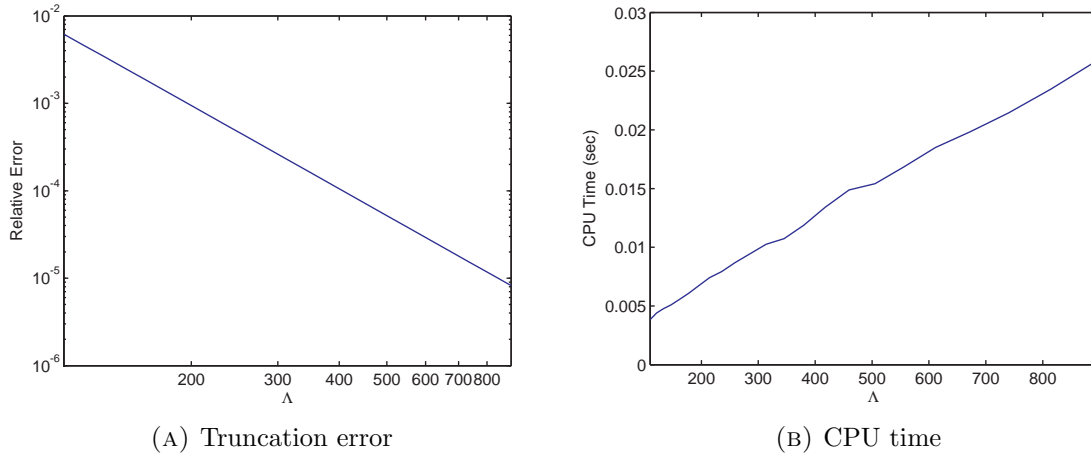


FIGURE 15. Truncation error (logarithmic plot) and CPU time under parabolic iFT, with $\alpha = 3$, for an ATM vanilla call option with $K = 1$, $\lambda_- = -11$, $\lambda_+ = 8$, $m_2 = 0.16$, $r = 0.03$, $q = 0$, $\tau = 0.004$, for varying Λ .

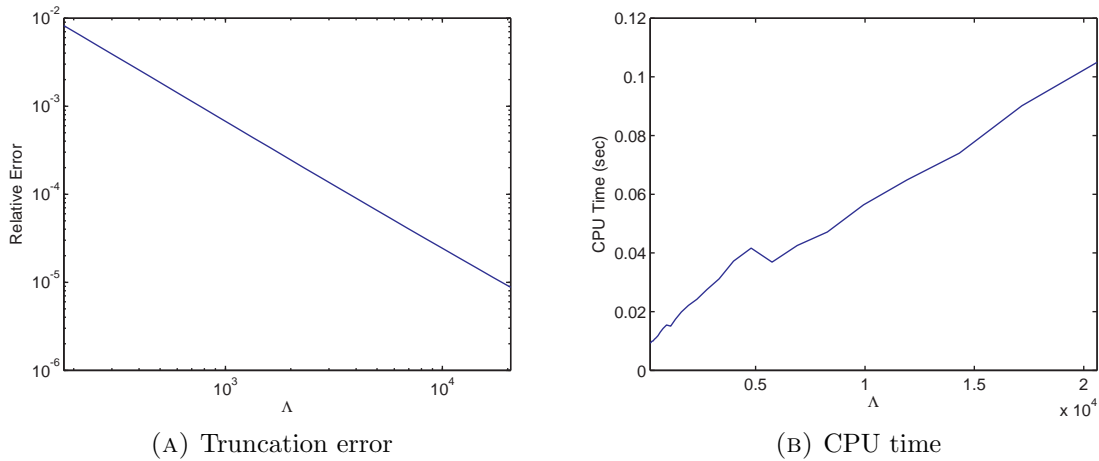


FIGURE 16. Truncation error (logarithmic plot) and CPU time under hyperbolic iFT, with $\alpha = 3.8$, for an ATM vanilla call option with $K = 1$, $\lambda_- = -11$, $\lambda_+ = 8$, $m_2 = 0.16$, $r = 0.03$, $q = 0$, $\tau = 0.004$, for varying Λ .

APPENDIX E. INTERPOLATION NEAR ATM FOR VANILLA OPTIONS

We interpolate the price of a vanilla call option at $x = -\mu\tau - \delta$ from either the prices or the prices and deltas¹⁷ on the grid $(x_j)_{j=0}^{N-1}$, for $N = 1, \dots, M$. We compare the performance of the following interpolation methods for $\Delta = 0.02$ and $\delta = 0.001$:

- a. Polynomial interpolation in the option prices (for $N = 2$ this corresponds to the usual linear interpolation).
- b. Polynomial interpolation, taking into account the option deltas as well as the prices, for $j = 0, \dots, N$.
- c. Polynomial interpolation, taking into account only the OTM option deltas, i.e. those for $j = 1, \dots, N$.
- d. Hermite spline interpolation in the option prices.
- e. Hermite spline interpolation, taking into account the option deltas as well as the prices for $j = 0, \dots, N$.
- f. Hermite spline interpolation, taking into account only the OTM option deltas, i.e. those for $j = 1, \dots, N$.

We do this for three separate sets of VG parameters. Each set has the same instantaneous second moment m_2 and different sets of steepness parameters.

Set A (medium steepness). $\lambda_- = -11$, $\lambda_+ = 8$, $m_2 = 0.16$,

Set B (large steepness). $\lambda_- = -30$, $\lambda_+ = 25$, $m_2 = 0.16$,

Set C (very large steepness). $\lambda_- = -50$, $\lambda_+ = 40$, $m_2 = 0.16$,

for $K = 1$, $r = 0.03$, $q = 0$, $\tau = 0.004$. In Figures 17–19 we plot the relative errors of the interpolated price at $x = -\mu\tau - \delta$ using polynomial interpolation versus the number of points N . We note that

1. Linear interpolation near ATM produces large errors.
2. Near ATM, the errors of polynomial interpolation usually blow up from around $N = 30$ onwards, when deltas are not used, or from around $N = 15 - 20$, when deltas are used. This is due to the Runge phenomenon, i.e. the large errors which occur at the ends of an interval when fitting high degree polynomials to a regular grid.
3. Near ATM, the minimum error when using ordinary polynomial interpolation is usually of the order of 1%. Taking OTM deltas into account, this error can generally be reduced, but it usually remains of the same order of magnitude.

¹⁷For the calculation of the delta using the ATMVG method, see Appendix F.

4. Including the ATM delta does not necessarily help. This effect, evident in the results obtained with Set A, is due to the fact that at $x = -\mu\tau$ the option price has a “kink” and the gamma is unbounded there if $\tau \in (0, 1/(2c)]$ (cf. Appendix F), as can be seen from Figures 20 and 21. For Set B and Set C, including the ATM delta does improve accuracy. The reason is that in these cases λ_- is very large in absolute value, and hence the probability density of upward jumps (which is most important for an OTM call option close to ATM) becomes much more regular near the origin, although, for Set B, it still diverges there.
5. Splines tend to perform worse than polynomial interpolation near ATM, since the option gamma is either unbounded or usually very large for $x \rightarrow -\mu\tau$.

To summarize, near the ATM point the relative errors of interpolation for vanilla options can usually be made as small as 0.5–1%, if a sufficiently high number of points N is used. The best results, for small to medium-sized $|\lambda_-|$ (for OTM calls) or λ_+ (for OTM puts) can be obtained by using polynomial interpolation in the ATM and OTM prices, and the OTM deltas.

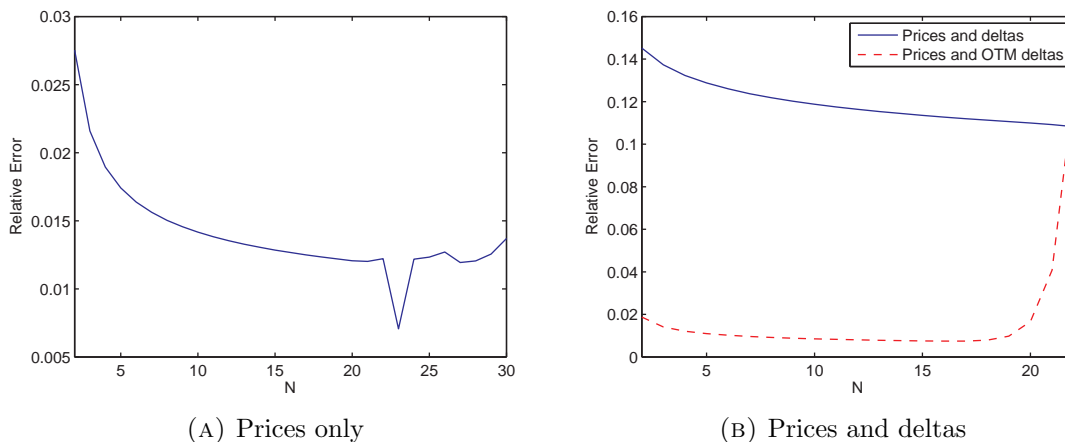


FIGURE 17. Relative errors of polynomial interpolation at $x = -\mu\tau - 0.001$ on a grid $x_j = -\mu\tau - 0.02j$, $j = 0, \dots, 31$. Call option with $\lambda_- = -11$, $\lambda_+ = 8$, $m_2 = 0.16$, $r = 0.03$, $q = 0$, $\tau = 0.004$, $K = 1$.

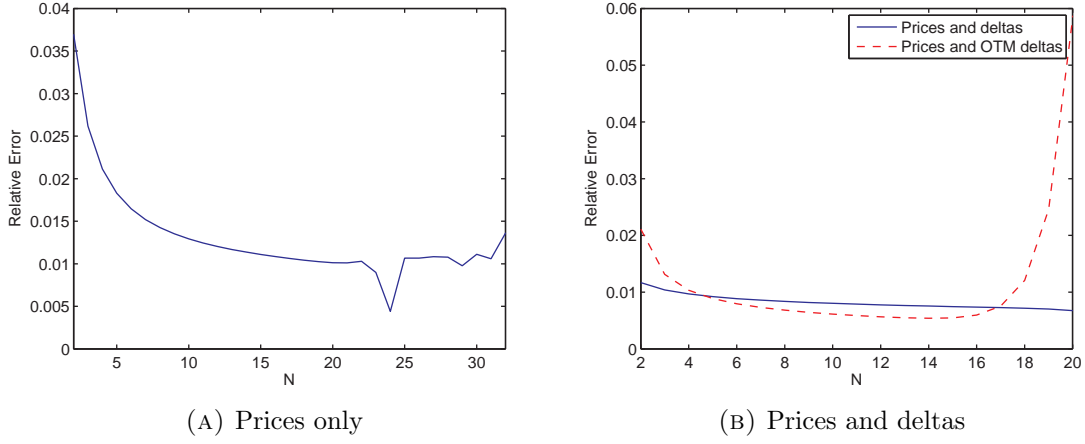


FIGURE 18. Relative errors of polynomial interpolation at $x = -\mu\tau - 0.001$ on a grid $x_j = -\mu\tau - 0.02j$, $j = 0, \dots, 31$. Call option with $\lambda_- = -30$, $\lambda_+ = 25$, $m_2 = 0.16$, $r = 0.03$, $q = 0$, $\tau = 0.004$, $K = 1$.

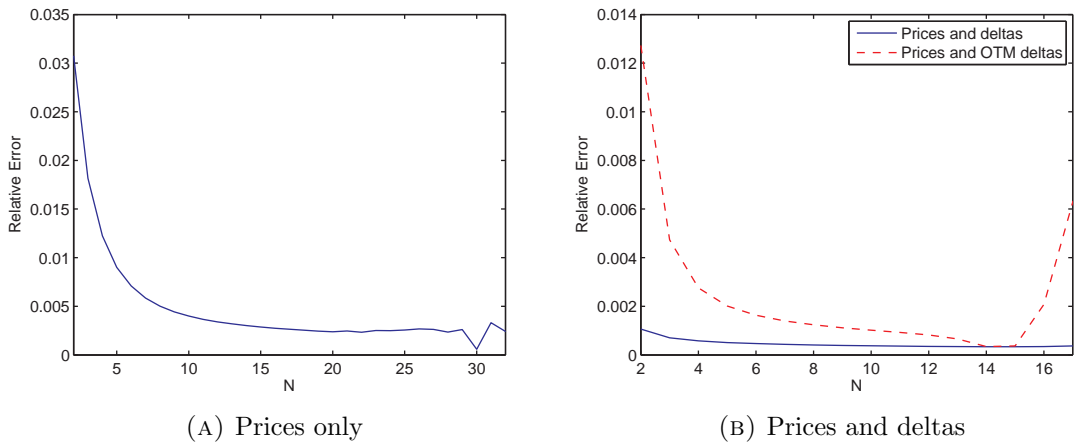


FIGURE 19. Relative errors of polynomial interpolation at $x = -\mu\tau - 0.001$ on a grid $x_j = -\mu\tau - 0.02j$, $j = 0, \dots, 31$. Call option with $\lambda_- = -50$, $\lambda_+ = 40$, $m_2 = 0.16$, $r = 0.03$, $q = 0$, $\tau = 0.004$, $K = 1$.

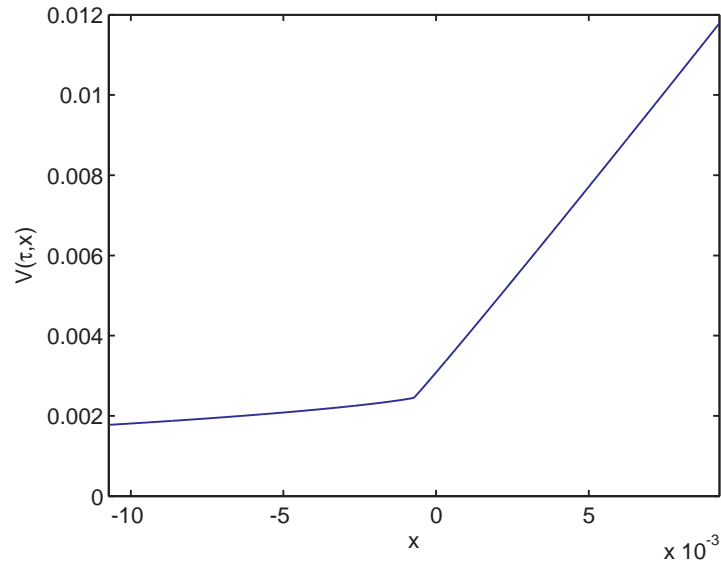


FIGURE 20. Call option price versus $x = \ln(S/K)$ for an ATM vanilla call option with $K = 1$, $\lambda_- = -11$, $\lambda_+ = 8$, $m_2 = 0.16$, $r = 0.03$, $q = 0$, $\tau = 0.004$. Note the “kink” at $x = -\mu\tau \simeq -7.22 \cdot 10^{-4}$.

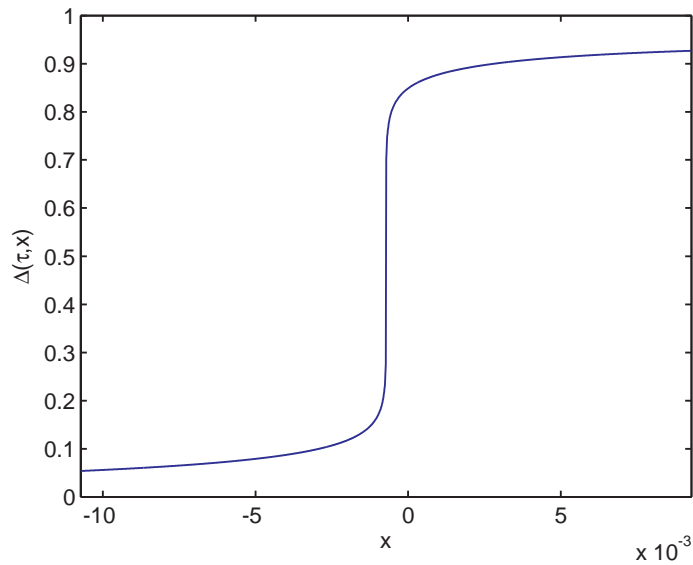


FIGURE 21. Call option delta versus $x = \ln(S/K)$ for an ATM vanilla call option with $K = 1$, $\lambda_- = -11$, $\lambda_+ = 8$, $m_2 = 0.16$, $r = 0.03$, $q = 0$, $\tau = 0.004$.

APPENDIX F. DELTAS, GAMMAS AND DIGITALS

Since $\partial_S = K^{-1}e^{-x}\partial_x$, the vanilla option delta $\Delta(\tau, x) = K^{-1}e^{-x}\partial_x V(\tau, x)$ is evidently given by

$$(F.1) \quad \Delta(\tau, x; \omega) = \frac{e^{-r\tau-x}}{2\pi} \int_{\text{Im}\xi=\omega} \frac{e^{ix\xi-\tau\psi(\xi)}}{i\xi-1} d\xi,$$

where $\Delta(\tau, x; \omega)$ is equal to the vanilla call delta $\Delta_c(\tau, x)$ if $\omega < -1$ and to the vanilla put delta $\Delta_p(\tau, x)$ if $\omega > -1$. From (F.1) we obtain

$$\Delta(\tau, x; \omega) = \frac{e^{-r\tau-(\omega+1)x}}{2\pi} \int_{-\infty}^{\infty} \frac{e^{ix'\eta-\tau\psi_{\omega}^0(\eta)}}{i\eta-\omega-1} d\eta.$$

Hence, in the ATMVG case, we can write

$$(F.2) \quad \Delta(\tau, -\mu\tau; \omega) = \frac{e^{-r'\tau-x}}{2\pi} \Phi(\sigma, \omega+1),$$

where $\Phi(\sigma, \omega+1)$ is given by (3.4). For the FastVG case, we have

$$(F.3) \quad \Delta(\tau, x; \omega) = \frac{e^{-r'\tau-\omega x'-x}}{\pi} \text{Re} \Psi(x'; \sigma, \omega+1),$$

where $\Psi(x'; \sigma, \omega+1)$ is given by (5.4).

For $\omega = -1$ and $\omega > -1$, (F.2)–(F.3) give the values of $\Delta_{-1}(\tau, x)$ (defined in the same way as $V_{-1}(\tau, x)$ in Subsection 3.3) and $\Delta_p(\tau, x)$, respectively. In each case, the vanilla call delta is given by

$$\Delta_c(\tau, x) = \Delta_{-1}(\tau, x) + \frac{1}{2}e^{-q\tau}, \Delta_c(\tau, x) = \Delta_p(\tau, x) + e^{-q\tau},$$

respectively

The payoff at maturity of a digital call or put option is given by $G(x) = \mathbb{1}_{[K, \infty)}(x)$ and $G(x) = \mathbb{1}_{(-\infty, K]}(x)$, respectively. Denote the price of the digital call by $V_{dc}(\tau, x)$ and that of the digital put by $V_{dp}(\tau, x)$. If, for $\omega \neq 0$, we define

$$(F.4) \quad V_d(\tau, x; \omega) = \frac{e^{-r\tau}}{2\pi} \int_{\text{Im}\xi=\omega} \frac{e^{ix\xi-\tau\psi(\xi)}}{i\xi} d\xi,$$

then $V_d(\tau, x; \omega) = V_{dc}(\tau, x)$ if $\omega < 0$ and $V_d(\tau, x; \omega) = -V_{dp}(\tau, x)$ if $\omega > 0$. In the ATMVG case, for $\omega \neq 0$, we can write

$$(F.5) \quad V_d(\tau, -\mu\tau; \omega) = \frac{e^{-r'\tau}}{2\pi} \Phi(\sigma, \omega),$$

and in the FastVG case

$$(F.6) \quad V_d(\tau, x; \omega) = \frac{e^{-r'\tau-\omega x'}}{\pi} \Psi(x'; \sigma, \omega).$$

The digital call and put prices are related by digital put-call parity: $V_{dc}(\tau, x) + V_{dp}(\tau, x) = e^{-r\tau}$. Regarding the case $\omega = 0$, with the same argument as in Subsection 3.3 it can easily be seen that $V_{dc}(\tau, x) + V_d(\tau, x; 0) = \frac{1}{2}e^{-r\tau}$, where $V_d(\tau, x; 0)$ is defined in terms of Cauchy's principal value. From this it follows, in particular, that for a symmetric VG process (with $\lambda_+ = -\lambda_-$), the ATM price of a digital call or put option is always equal to $\frac{1}{2}e^{-r\tau}$. For $x' \neq 0$ and $\omega \in [-1, 0]$, the method of Subsection 5.2 can be applied.

For gamma, we have the following result.

Proposition F.1. *The non-ATM gamma of a vanilla call or put option under VG* $\text{Gamma}(\tau, x) = K^{-1}e^{-x}\partial_x\Delta(\tau, x)$ *is given by*

$$(F.7) \quad \text{Gamma}(\tau, x) = \frac{e^{-r\tau-2x}}{2\pi K} \int_{\text{Im}\xi=\omega} e^{ix\xi-\tau\psi(\xi)} d\xi,$$

for any $\omega \in \mathbb{R}$.

Proof. By changing variable in (F.1), so that $\xi \mapsto \xi - i$, we obtain

$$(F.8) \quad \begin{aligned} \Delta(\tau, x; \omega) &= \frac{e^{-r\tau}}{2\pi} \int_{\text{Im}\xi=\omega-1} \frac{e^{ix\xi-\tau\psi(\xi-i)}}{i\xi} d\xi \\ &= \frac{e^{-r\tau}}{2\pi} \int_{\text{Im}\xi=\omega-1} \frac{e^{ix'\xi-\tau\tilde{\psi}_0(\xi)}}{i\xi} d\xi, \end{aligned}$$

where $\tilde{\psi}_0(\xi) = -\mu + \psi_0(\xi - i)$, and $\psi_0(\xi) = i\mu\xi + \psi(\xi)$. We are not allowed to differentiate under the integral sign in (F.8), since the integrand only decays like $|\xi|^{-2c\tau-1}$ for $\xi \rightarrow \infty$. Noting that $e^{ix'\xi}d\xi = (ix')^{-1}de^{ix'\xi}$, we can integrate by parts. For $x' \neq 0$, we have

$$\begin{aligned} \Delta(\tau, x; \omega) &= -\frac{e^{-r\tau}}{2\pi ix'} \int_{\text{Im}\xi=\omega-1} e^{ix'\xi} \partial_\xi \left(\frac{e^{-\tau\tilde{\psi}_0(\xi)}}{i\xi} \right) d\xi \\ &= \frac{e^{-r\tau}}{2\pi x'} \int_{\text{Im}\xi=\omega-1} e^{ix'\xi} \left(\xi^{-1} \partial_\xi e^{-\tau\tilde{\psi}_0(\xi)} - \xi^2 e^{-\tau\tilde{\psi}_0(\xi)} \right) d\xi. \end{aligned}$$

This expression can be differentiated w.r.t. x under the integral sign, since the integrand now decays like $|\xi|^{-2c\tau-2}$. For $x' \neq 0$ we can write

$$(F.9) \quad \begin{aligned} \partial_x \Delta(\tau, x; \omega) &= -\frac{e^{-r\tau}}{2\pi x'^2} \int_{\text{Im}\xi=\omega-1} e^{ix'\xi} \xi^{-1} \partial_\xi e^{-\tau\tilde{\psi}_0(\xi)} d\xi + \frac{e^{-r\tau}}{2\pi x'^2} \int_{\text{Im}\xi=\omega-1} e^{ix'\xi} \xi^2 e^{-\tau\tilde{\psi}_0(\xi)} d\xi \\ &\quad - \frac{e^{-r\tau}}{2\pi ix'} \int_{\text{Im}\xi=\omega-1} e^{ix'\xi} \partial_\xi e^{-\tau\tilde{\psi}_0(\xi)} d\xi + \frac{e^{-r\tau}}{2\pi ix'} \int_{\text{Im}\xi=\omega-1} e^{ix'\xi} \xi^{-1} e^{-\tau\tilde{\psi}_0(\xi)} d\xi. \end{aligned}$$

Integrating by parts in the third term, we have

$$-\frac{e^{-r\tau}}{2\pi ix'} \int_{\text{Im}\xi=\omega-1} e^{ix'\xi} \partial_\xi e^{-\tau\tilde{\psi}_0(\xi)} d\xi = \frac{e^{-r\tau}}{2\pi} \int_{\text{Im}\xi=\omega-1} e^{ix'\xi-\tau\tilde{\psi}_0(\xi)} d\xi.$$

Hence, it remains to show that the first, second and fourth terms in (F.9) sum to zero. Integrating by parts in the first term, we obtain

$$\begin{aligned} & -\frac{e^{-r\tau}}{2\pi x'^2} \int_{\text{Im } \xi = \omega - 1} e^{ix'\xi} \xi^{-1} \partial_\xi e^{-\tau\tilde{\psi}_0(\xi)} = \frac{e^{-r\tau}}{2\pi x'^2} \int_{\text{Im } \xi = \omega - 1} e^{-\tau\tilde{\psi}_0(\xi)} \partial_\xi (\xi^{-1} e^{ix'\xi}) d\xi \\ & = -\frac{e^{-r\tau}}{2\pi ix'} \int_{\text{Im } \xi = \omega - 1} e^{-\tau\tilde{\psi}_0(\xi)} \xi^{-1} e^{ix'\xi} d\xi - \frac{e^{-r\tau}}{2\pi x'^2} \int_{\text{Im } \xi = \omega - 1} e^{-\tau\tilde{\psi}_0(\xi) - \tau\psi(\xi)} \xi^{-2} d\xi. \end{aligned}$$

Substituting this expression for the first term on the RHS of (F.9), it can be seen that all terms in (F.9) cancel each other out, apart from the third one. Equation (F.7) is obtained by changing the variable back: $\xi \mapsto \xi + i$. \square

Finally, we note the following result from [5] about the limit of the vanilla gamma for $x \rightarrow -\mu\tau$.

Proposition F.2 (Proposition 7.2, [5]). *Let X be a VG process. Then, for $\tau \in (0, 1/(2c)]$, $\text{Gamma}(\tau, x) \rightarrow +\infty$ as $x \rightarrow -\mu\tau$.*

APPENDIX G. NUMERICAL CALCULATION OF THE HILBERT TRANSFORM

The method, employed in [14], approximates the Hilbert transform of a function f by the multiplication of the vector of function values \vec{f} , of dimension M , by a Toeplitz matrix T . Since it is known that any $M \times M$ Toeplitz matrix can be embedded into a wider circulant matrix C , and that the multiplication of a circulant matrix C by a vector \vec{x} can be realized as follows¹⁸ [10]

$$C\vec{x} = \text{ifft}(\text{fft}(\vec{c}) .* \text{fft}(\vec{x}));$$

where \vec{c} is first column of C , we end up with the following algorithm for the calculation of the discrete Hilbert transform

- i. Compute the entries of the vector $\vec{t} = (t_j)_{j=1}^M$, given by

$$t_j = \frac{1 - (-1)^{1-j}}{\pi(1-j)},$$

for $k = 2, \dots, M$, and $t_1 = 0$.

- ii. Let N denote the smallest power of 2 such that $N \geq 2M - 1$. If M is already a power of 2, then $N = 2M$.
- iii. Construct the vector \vec{c} such that

$$\vec{c} = [-\vec{t} \text{ zeros}(1, N - 2 * M + 1) \vec{t}(\text{end} : -1 : 2)];$$

- iv. Augment the vector \vec{f} by appending $N - M$ zeros to it

$$\tilde{f} = [\vec{f} \text{ zeros}(1, N - M)];$$

¹⁸In this appendix, we use a notation reminiscent of MATLAB syntax.

v. Calculate

$$\tilde{g} = \text{ifft} \left(\text{fft}(\vec{c}) \cdot * \text{fft}(\tilde{f}) \right);$$

vi. Finally, discard the last $N - M$ elements in the vector \tilde{g} to obtain the result vector

$$\vec{g} = \tilde{g}(1 : M);$$

We note that the vector $\text{fft}(\vec{c})$ can be computed in advance, for several values of M , therefore this method only involves the calculation of one FFT and one inverse FFT of dimensions $2M$ each.

APPENDIX H. DISCRETIZATION ERROR BOUND FOR $\Psi_1(x; \sigma, 0; \Lambda)$.

Due to the presence of the indicator function in the expression (5.23) for $\psi(\eta)$, we need not concern ourselves with the truncation error in (5.22). The following result can be used to choose an appropriate value of the discretization mesh h in order to ensure that the error lies within a specified tolerance.

Proposition H.1. *Denote by $\tilde{\Psi}_1(x; \sigma, 0; \Lambda; h)$ the approximation to $\Psi_1(x; \sigma, 0; \Lambda)$ calculated according to (5.22) using the method of Appendix G, and define the discretization error with mesh h as*

$$\text{Disc.Err.}(\Psi_1; x; \sigma, 0; \Lambda; h) = |\tilde{\Psi}_1(x; \sigma, 0; \Lambda; h) - \Psi_1(x; \sigma, 0; \Lambda)|.$$

Then, if $c\tau < 1$, then the discretization error satisfies the following bounds¹⁹

(H.1)

$$\begin{aligned} \text{Disc.Err.}(\Psi_1; x; \sigma, 0; \Lambda; h) &\leq \mu(\chi, h)(e^{\sigma x} + e^{-\sigma x}) \frac{\pi(2\sigma)^{-c\tau} \Lambda^{1-c\tau} {}_2F_1\left(\frac{1-c\tau}{2}, \frac{c\tau}{2}, \frac{3-c\tau}{2}, -\frac{\Lambda^2}{4\sigma^2}\right)}{1-c\tau} \\ &\leq \mu(\chi, h)(e^{\sigma x} + e^{-\sigma x}) \frac{\pi(2\sigma)^{-c\tau} \Lambda^{1-c\tau}}{1-c\tau}, \end{aligned}$$

where $\mu(\chi, h) = e^{-2\pi\chi/h} / (1 - e^{-2\pi\chi/h})$.

Proof. The discretization error in the numerical calculation of the Hilbert transform of a function f is bounded above by [14]

$$\frac{e^{-2\pi\chi/h}}{1 - e^{-2\pi\chi/h}} \|\psi\|_{H^1(\mathcal{D}_\chi)},$$

where h is the mesh of the discretization grid, $\mathcal{D}_\chi := \{z \in \mathbb{C} : |\text{Im } z| < \chi\}$, and $H^1(\mathcal{D}_\chi)$ denotes the Hardy space of functions f analytic in the strip \mathcal{D}_χ such that $\int_{-\chi}^\chi f(x + iy) dy \rightarrow 0$ as $x \rightarrow \pm\infty$, and the Hardy norm is finite

$$\|f\|_{H^1(\mathcal{D}_\chi)} := \lim_{y \rightarrow \chi^-} \left[\int_{\mathbb{R}} |f(x + iy)| dx + \int_{\mathbb{R}} |f(x - iy)| dx \right] < \infty.$$

¹⁹For typical parameter ranges, the condition $c\tau < 1$ is almost always satisfied when FastVG is used to calculate the price or delta of a vanilla option, or the price of a digital option.

In our case $\chi = \sigma$, and

$$\begin{aligned} \int_{\mathbb{R}} |\psi(\eta + i\omega)| d\eta &= \int_{-\Lambda}^{\Lambda} |e^{ix(\eta+i\omega)}(\sigma^2 + (\eta + i\omega)^2)^{-c\tau}| d\eta \leq e^{-\omega x} \int_{-\Lambda}^{\Lambda} \frac{d\eta}{|\sigma^2 + (\eta + i\omega)^2|^{c\tau}} \\ &= 2e^{-\omega x} \int_0^{\Lambda} \frac{d\eta}{|\sigma^2 + \eta^2 - \omega^2 + 2i\omega\eta|^{c\tau}} \\ &= 2e^{-\omega x} \int_0^{\Lambda} \frac{d\eta}{[(\sigma^2 - \omega^2 + \eta^2)^2 + 4\omega^2\eta^2]^{c\tau/2}}. \end{aligned}$$

Define

$$H(\omega, \sigma, \Lambda, c\tau) = \int_0^{\Lambda} \frac{d\eta}{[(\sigma^2 - \omega^2 + \eta^2)^2 + 4\omega^2\eta^2]^{c\tau/2}}.$$

Then the norm $\|\psi\|_{H^1(\mathcal{D}_\sigma)}$ admits the bound

$$\|\psi\|_{H^1(\mathcal{D}_\sigma)} \leq 2e^{\sigma x} H(-\sigma, \sigma, \Lambda, c\tau) + 2e^{-\sigma x} H(\sigma, \sigma, \Lambda, c\tau),$$

and (H.1) follows from

$$H(\sigma, \sigma, \Lambda, c\tau) = H(-\sigma, \sigma, \Lambda, c\tau) = \frac{(2\sigma)^{-c\tau} \Lambda^{1-c\tau} {}_2F_1\left(\frac{1-c\tau}{2}, \frac{c\tau}{2}, \frac{3-c\tau}{2}, -\frac{\Lambda^2}{4\sigma^2}\right)}{1-c\tau},$$

which can be proved using (A.3). The bound (H.2) can easily be proved by discarding the η^4 term in the denominator of $H(\sigma, \sigma, \Lambda, c\tau)$. \square

REFERENCES

- [1] M. Abramowitz and I. Stegun (eds.). *Handbook of Mathematical Functions, with Formulas, Graphs and Mathematical Tables*. Dover Publications, Mineola, NY, 1965.
- [2] S.I. Boyarchenko and S.Z. Levendorskii. On rational pricing of derivative securities for a family of non-gaussian processes. Preprint 98/7, Institut für Mathematik, Universität Potsdam, 1998. Available at: <http://opus.kobv.de/ubp/volltexte/2008/2519/>.
- [3] S.I. Boyarchenko and S.Z. Levendorskii. Option pricing for truncated Lévy processes. *International Journal of Theoretical and Applied Finance*, 3(3):549–552, July 2000.
- [4] S.I. Boyarchenko and S.Z. Levendorskii. *Non-Gaussian Merton-Black-Scholes Theory*, volume 9 of *Adv. Ser. Stat. Sci. Appl. Probab.* World Scientific Publishing Co., River Edge, NJ, 2002.
- [5] S.I. Boyarchenko and S.Z. Levendorskii. New efficient versions of Fourier transform method in applications to option pricing. Working paper, May 2011. Available at SSRN: <http://papers.ssrn.com/abstract=1846633>.
- [6] P. Carr, H. Geman, D.B. Madan, and M. Yor. The fine structure of asset returns: an empirical investigation. *Journal of Business*, 75:305–332, 2002.
- [7] P. Carr, A. Hogan, and H. Stein. Time for a change: The Variance Gamma model and option pricing. Working paper, January 2007. Available at SSRN: <http://papers.ssrn.com/abstract=956625>.
- [8] P. Carr and D. Madan. Saddlepoint methods for option pricing. *Journal of Computational Finance*, 13(1):49–61, Fall 2009.

- [9] P. Carr and D.B. Madan. Option valuation using the Fast Fourier Transform. *Journal of Computational Finance*, 2(4):61–73, 1999.
- [10] P.J. Davis. *Circulant Matrices*. Chelsea Publishing Company, New York, 2nd edition, 1994.
- [11] A. Eydeland. A fast algorithm for computing integrals in function spaces: financial applications. *Computational Economics*, 7:277–285, 1994.
- [12] F. Fang and C.W. Oosterlee. A novel pricing method for european options based on Fourier-cosine series expansions. *SIAM Journal on Scientific Computing*, 31(2):826–848, 2008.
- [13] F. Fang and C.W. Oosterlee. Pricing early-exercise and discrete barrier options by Fourier-cosine series expansions. *Numerische Mathematik*, 114(1):27–62, 2009.
- [14] L. Feng and V. Linetsky. Pricing discretely monitored barrier options and defaultable bonds in Lévy process models: a fast Hilbert transform approach. *Mathematical Finance*, 18(3):337–384, July 2008.
- [15] W. Gander and W. Gautschi. Adaptive quadrature – revisited. *BIT*, 40(1):84–101, March 1993.
- [16] S. Heston. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *Review of Financial Studies*, 6:327–343, 1993.
- [17] S. Levendorskiĭ and J. Xie. Fast pricing and calculation of sensitivities of OTM European options under Lévy processes. *Journal of Computational Finance*. Available at SSRN: <http://papers.ssrn.com/abstract=1589809>.
- [18] R. Lord and C. Kahl. Optimal Fourier inversion in semi-analytical option pricing. *Journal of Computational Finance*, 10(4):1–30, Summer 2007.
- [19] D.B. Madan and E. Seneta. The Variance Gamma (V.G.) model for share market returns. *Journal of Business*, 63:511–524, 1990.
- [20] D.P. Madan, P. Carr, and E.C. Chang. The Variance Gamma Process and Option Pricing. *European Finance Review*, 2(79):79–105, 1998.
- [21] B. Mandelbrot. The variation of certain speculative prices. *Journal of Business*, 36(4):394–419, October 1963.
- [22] K. Pearson, G.B. Jeffrey, and E.M. Elderton. On the distribution of the first product-moment coefficient, in samples drawn from an indefinitely large normal population. *Biometrika*, 21(1-4):164–193, December 1929.
- [23] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes in C*. Cambridge University Press, New York, 2nd edition, 1992.
- [24] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes – The Art of Scientific Computing*. Cambridge University Press, New York, 3rd edition, 2007.
- [25] L.F. Shampine. Vectorized adaptive quadrature in MATLAB. *Journal of Computational and Applied Mathematics*, 211(2):131–140, January 2008.
- [26] N. M. Temme. Computational aspects of incomplete gamma functions with large complex parameters. In R. V. M. Zahar, editor, *Approximation and Computation. A Festschrift in Honor of Walter Gautschi.*, volume 119 of *International Series of Numerical Mathematics*, pages 551–562. Birkhäuser Boston, Boston, MA, 1994.
- [27] G.N. Watson. *Theory of Bessel Functions*. Cambridge University Press, Cambridge, 1922.