Scientific Research Publishing

Journal of Mathematical Finance, 2016, 6, 810-841
http://www.scirp.org/journal/jmf
ISSN Online: 2162-2442
ISSN Print: 2162-2434

Pricing Asian Options: A Comparison of Numerical and Simulation Approaches Twenty Years Later

Akos Horvath[1], Peter Medvegyev[2]

[1]Department of Finance, University of Vienna, Vienna, Austria
[2]Department of Mathematics, Corvinus University of Budapest, Budapest, Hungary
Email: akos.horvath@univie.ac.at, medvegyev@uni-corvinus.hu

How to cite this paper: Horvath, A. and Medvegyev, P. (2016) Pricing Asian Options: A Comparison of Numerical and Simulation Approaches Twenty Years Later. *Journal of Mathematical Finance*, **6**, 810-841.
http://dx.doi.org/10.4236/jmf.2016.65056

Received: August 29, 2016
Accepted: November 15, 2016
Published: November 18, 2016

Abstract

The calculation of the Asian option value has posed a great challenge to financial mathematicians as well as practitioners for the last two decades. Since there exists no analytical valuation formula to date, one has to resort to other methods to price this commonly used derivative product. One possibility is the usage of simulation approaches, which however are especially inefficient for Asian options, due to their dependence on the entire stock price trajectory. Another alternative is resorting to semi-analytical methods, based on the inversion of the option price's Laplace transform, which however are prone to severe numerical difficulties. In this paper, we seek answer to the question whether it is possible to improve on the efficiency of the semi-analytical approach, implementing and comparing different numerical algorithms, so that they could be applied in real-life situations. We look into whether today's superior computer environment has changed the relative strength of numerical and simulation approaches with regards to Asian option pricing. Based on a comprehensive analysis of speed and reliability, we find that the Laplace transform inversion method can be further enhanced, pushing down the prior critical value from 0.01 to 0.005 and the calculation time from 20 - 30 seconds to 3 - 4 seconds. This renders the numerical approach readily applicable for practitioners; however, we conclude that the simulation approach is a more efficient option when $\sigma^2 T < 0.01$.

Keywords

Asian Options, Laplace Transform Inversion, Monte Carlo Simulation

1. Introduction

Asian options are popular hedging instruments in the hands of financial risk managers, owing to their special payout structure and cost-effectiveness. With regard to exercise

behavior, these options are "European", in the sense that they can be exercised only at a pre-agreed (maturity) date; however, as opposed to (plain) vanilla products, Asian options have an "exotic" payout at maturity

$$f\left(T,S\left(T\right)\right)=\left(Avg_{[0,T]}\left(S\right)-K\right)^{+}, \tag{1}$$

where K stands for the pre-agreed strike price and $Avg_{[0,T]}\left(S\right)$ stands for the time average of the underlying asset's prices from time 0 to time T (*i.e.* during the lifetime of the option)[1]. The above equation implies that Asian options are path-dependent; moreover, since their value depends on the average of all underlying prices during the option's lifetime, they may be considered as the quintessence of path-dependent derivatives.

The exotic nature of this option type has important financial and mathematical implications for us. From a financial (or practical) perspective, the stabilizing effect of averaging in Equation (1) makes Asian options a cheaper, yet more reliable alternative to their vanilla counterparts for financial risk managers. By taking the average of market prices for a given time period, the adverse effects of potential market manipulation and price jumps are minimized, which renders these options a cost-effective hedging instrument, especially in market environments where either the volume is low (e.g. in the corporate bond or commodity markets), or the volatility is high (e.g. in the currency and interest rate markets). As a result, according to a survey including 200 derivative-using non-financial U.S. firms conducted by Bodnar *et al.* [1], Asian options are the most popular exotic payout options chosen by non-financial firms for risk management.

From a mathematical (or theoretical) perspective, the valuation of Asian options is particularly challenging even in the most simple pricing models, which is partly due to path-dependence and partly because the probability distribution of averages is usually unknown. The average itself is defined as either the arithmetic or the geometric mean of the underlying asset's prices; the former being more common on the market, while the latter having more convenient mathematical properties.

In the classical Black-Scholes framework, the value of the geometric-average Asian option can be determined just as easily (for the closed-form formula refer to Turnbull *et al.* [2]) as in the vanilla case, since the product of the lognormally distributed asset prices follows lognormal distribution as well. In contrast, the valuation of the arithmetic-average Asian option poses a far greater challenge than its geometric counterpart, and a closed-form formula has not been derived to date. Nevertheless, in their work Geman and Yor [3] derive an analytical expression for the Laplace transform of the so-called "normalized" Asian option price, making it possible to determine the option value by means of numerical (inversion) methods. At the same time, Geman and Eydeland [4] find that these methods are intractable for small values of $\sigma^2 T$.

Another approach to pricing arithmetic-average Asian options is using Monte Carlo simulation, which is a far more robust, yet computationally demanding method. Em-

[1]Note that the maturity date and the end of the averaging period need not coincide. In practice, however, they usually do, and thus no distinction is made in this article.

ploying the geometric-average Asian option price as control variate, Fu *et al.* [5] make a comparison between the two approaches (*i.e.* between numerical inversion methods and control variate Monte Carlo methods), and find that difficulties with the Laplace transform arise when $\sigma^2 T$ is lower than 0.01. Below this value, the inversion algorithms employed fail to converge, and one has to resort to the simulation approach or make approximations (e.g. by interpolation). This is an especially unwelcome result, as $\sigma^2 T$ is often smaller than this critical value under usual market circumstances, thus rendering the numerical approach impractical.

In this paper, we seek answer to two important questions naturally arising from the results above. Considering the development in computational power,

1) Is it possible to find a more efficient algorithm than the "Euler method" employed by Fu *et al.* [5]? If so, how does the critical value of convergence change for the numerical approach?

2) What is an optimal daily read-replication number combination for the Monte Carlo method, with regard to both discretization and simulation error? How has the relation of the different approaches changed in the last ten years?

In Section 2, we introduce the basic principles of Asian option (and in general, financial derivative) pricing, making a clear distinction between the different valuation approaches. Then, in Section 3, we proceed with an empirical analysis, comparing the performance of pricing algorithms in MATLAB®. Finally, we make recommendations on the best valuation practices, and conclude.

2. Valuation Methodology

In order to price Asian options, it is necessary to agree on the specific risk-neutral framework used, which is the Black-Scholes model in this paper. In this section, first we briefly recapitulate the assumptions and results of this model, then we introduce the risk-neutral valuation logic and make a distinction between the main pricing methods. Finally, we apply these to the particular valuation problem of Asian options, as well as discuss what computational difficulties arise with the different algorithms.

2.1. The Black-Scholes Model

In their original model, Black and Scholes [6] make the following assumptions about the behavior of asset prices, interest rates and financial markets:

- the risk-free interest rate is constant over time (as opposed to models in which interest rates are deterministic or stochastic processes);
- the stock price (of any firm) follows geometric Brownian motion "with a variance rate proportional to the square of the stock price";
- the option is a European-exercise option;
- the stock pays no dividends (or other interim cash flows);
- there are no transaction costs of trading or borrowing;
- there are no limitations to trading (including the ones concerning short-selling or the trading of fractional quantities);

- there are no limitations to taking out a loan or making a deposit at the risk-free interest rate.

Although the last few assumptions are rather technical and only ensure that the functions and processes are well-behaved, the first three assumptions form the core of the model and define very specific stock price and interest rate behavior. There are effectively three financial products in this setup:

- a risk-free bond B,
- a stock S, whose price follows geometric Brownian motion,
- and a European-exercise derivative product f, whose price is a real function of the stock's price,

which, if looked at as processes, have the respective price dynamics

$$\mathrm{d}B(t) \triangleq Br\mathrm{d}t$$

$$\mathrm{d}S(t) \triangleq \mu S(t)\mathrm{d}t + \sigma S(t)\mathrm{d}W(t)$$

$$\mathrm{d}f\left(t, S(t)\right) \triangleq \left(\frac{\partial f}{\partial t} + \frac{\partial f}{\partial S}\mu S(t) + \frac{1}{2}\frac{\partial^2 f}{\partial S^2}\sigma^2 S^2(t)\right)\mathrm{d}t + \frac{\partial f}{\partial S}\sigma S(t)\mathrm{d}W(t),$$

where W stands for the standard Wiener process. It follows (from Itô's lemma) that the logarithm of S is a normally distributed variate having the dynamics

$$\mathrm{d}X(t) \triangleq d\ln\left(S(t)\right) = \left(\mu - \frac{1}{2}\sigma^2\right)\mathrm{d}t + \sigma\mathrm{d}W(t). \tag{2}$$

Based on the assumptions above, using a replication method called dynamic delta hedging, Black and Scholes [6] derive the renowned partial differential equation (PDE) for the derivative price

$$rf(t) = \frac{\partial f}{\partial t} + r\frac{\partial f}{\partial S}S(t) + \frac{1}{2}\frac{\partial^2 f}{\partial S^2}\sigma^2 S^2(t), \tag{3}$$

which is a notable result, since in many cases it can be solved by means of standard PDE methods.

2.2. Risk-Neutral Valuation

Having chosen the specific price dynamics of asset prices, in theory we can calculate the expected value of the option's payout at maturity, which, if discounted at the proper rate of return, yields the option's value. Alas, this rate of return is unknown in most cases, moreover, it tends to change during the derivative product's lifetime, rendering the above pricing logic unusable.

Fortunately, there exists an elegant financial mathematical trick called risk-neutral valuation, which makes it possible to discount the expected value of the option's payout at the (known) risk-free interest rate. In what follows we briefly recapitulate the tenets of this logic, then we show how it is applied in practice.

Notation 2.1. *Let* $\left(\Omega, \mathcal{F}, \{\mathcal{F}_t\}_{t\geq 0}, \mathbb{P}\right)$ *be a filtered probability space, where* $\{\mathcal{F}_t\}_{t\geq 0}$ *is the natural filtration generated by Wiener process* $W_{\mathbb{P}}$.

Definition 2.2. *A* $\pi(t) = \left(\beta(t), \gamma(t)\right)$ *process pair is called investment strategy if*

both processes are predictable with respect to $\mathcal{F}_t$.

Definition 2.3. *An investment strategy is self-financing* (*SF*) *if* $\forall t$: $B(t)\mathrm{d}\beta(t) + S(t)\mathrm{d}\gamma(t) = 0$ *holds.*

Notation 2.4. *Let* V_π *be the value process of the portfolio* (*determined by a given* π *investment strategy*) *which contains* $\beta(t)$ *bonds and* $\gamma(t)$ *stocks at any given time. Accordingly, let* $\tilde{V}_\pi = \frac{V_\pi}{B}$ *be the discounted value process of the portfolio.*

It can be shown that given the right model conditions (*i.e.* no arbitrage, market completeness), the dynamics of the stock price could be expressed as

$$\mathrm{d}S(t) = rS(t)\mathrm{d}t + \sigma S(t)\mathrm{d}W_{\mathbb{Q}}(t),$$

with respect to a unique probability measure $\mathbb{Q}$. It follows that the discounted value process $\tilde{V}_\pi$ of any portfolio determined by a given π self-financing investment strategy will be a $\mathbb{Q}$ martingale, implying

$$E_{\mathbb{Q}}\left(\tilde{V}_\pi(t) \mid \mathcal{F}_s\right) = \tilde{V}_\pi(s) \qquad \forall s \le t.$$

Applying the martingale representation theorem, it can be shown that for any European-exercise derivative there exists a corresponding self-financing investment strategy π^* replicating the value process of a particular derivative. In preparation for the final step, let us define a "risk-neutral stock"

$$\mathrm{d}S^{(r)}(t) \triangleq rS^{(r)}(t)\mathrm{d}t + \sigma S^{(r)}(t)\mathrm{d}W_{\mathbb{P}}(t), \tag{4}$$

noting that the distribution of $S^{(r)}$ with respect to measure $\mathbb{P}$ is the same as the distribution of S with respect to measure $\mathbb{Q}$, which is summarized in the following theorem.

Theorem 2.5. (Risk-Neutral Valuation) *The present value of a European-exercise derivative equals to the expectation of the derivative price with respect to probability measure* $\mathbb{Q}$ *discounted at the risk-free interest rate. By assuming that the stock price has risk-neutral dynamics as defined in Equation* (4), *this expectation can be calculated with regards to probability measure* $\mathbb{P}$.

$$f\left(0, S(0)\right) = \mathrm{e}^{-rt}E_{\mathbb{Q}}\left(f\left(t, S(t)\right)\right) = \mathrm{e}^{-rt}E_{\mathbb{P}}\left(f\left(t, S^{(r)}(t)\right)\right) \tag{5}$$

Having described the fundamental logic of risk-neutral valuation, let us now turn to introducing the different pricing methods, which apply the above principles in practice.

2.2.1. Analytical Methods

Analytical methods aim at deriving a closed-form expression for the price of the particular derivative. This could be attempted by

- either solving Equation (3) with the appropriate boundary conditions given by the payout of the derivative at expiry,
- or calculating the risk-neutral expected value of the derivative at expiry, using the Risk-Neutral Valuation principle outlined in Section 2.2.

Whichever method we choose, the existence of a closed-form expression is not guaranteed, depending on the payout structure of the derivative. Hence, the valuation

problem becomes purely mathematical and even though in some cases the derivation of a pricing formula is extremely challenging, financial mathematicians put substantial effort into it. The reason for this is that a closed-form expression for a particular financial derivative's price yields an easy-to-calculate value, rendering analytical methods computationally superior to other (e.g. numerical or simulation) approaches. Nonetheless, apart from a few notable examples (e.g. vanilla options or geometric-average Asian options), it is not possible to derive a pricing formula. For instance, there has not been one found yet in the case of arithmetic-average Asian options.

2.2.2. Numerical Methods

Although a closed-form solution often proves to be elusive, sometimes it is possible to derive a complex but deterministic (semi-analytical) pricing formula. For instance, we may arrive at a complex (single/double/triple) integral form, which can be approximated by means of numerical methods (e.g. quadratures). As we explain in Section 2.3, this is the case for arithmetic-average Asian options, where only the Laplace transform of the option price could be determined in closed-form and the option price itself has to be calculated using numerical inversion techniques.

2.2.3. Simulation Methods

As an alternative to analytical and numerical methods, we can turn to simulation techniques known as "Monte Carlo methods". There exist numerous approaches but the basic principle is the same (see Kalos and Whitlock [7]):

1) Using random number generating algorithms, generate outcomes for a random variate with known distribution.

2) Applying the Law of Large Numbers, estimate the unknown expected value with the average of the generated outcomes.

3) When applicable, increase the sample size, so as to constrain the estimate into a desired confidence interval.

Monte Carlo methods are computationally intensive but easy to understand and flexible algorithms in the sense that they are compatible with a broad family of (physical, mathematical or financial) stochastic models. From the perspective of financial derivative valuation, the flexibility of Monte Carlo simulation allows for complex stock price movements (e.g. jump diffusion) and derivative payouts (e.g. path-dependency or early exercise), which are otherwise extremely hard to handle by analytical methods.

Without going into further details on the specific algorithms used in Section 3, we must point out the notoriously slow convergence of simulation methods. As stated earlier, these techniques rely heavily on the Law of Large Numbers, estimating the expected value by the average of simulated outcomes, which implies that the standard deviation of the estimate will be proportional to $n^{-0.5}$. In practice this means that for every additional decimal digit precision, we have to make a hundred times greater simulation, which means that the traditional Monte Carlo method becomes computationally intractable after the first 4 - 6 decimal digits of precision improvement.

For the above mentioned reasons, it is necessary to employ advanced simulation

techniques so as to reduce the variance of the estimate. Without aiming at completeness, we list a few variance reduction techniques here (for a detailed description refer to Kleijnen *et al.* [8]):

- common random numbers (including antithetic variates);
- control variates;
- conditioning and stratified sampling;
- importance sampling and splitting;
- quasi (not fully random) Monte Carlo simulation.

Depending on the statistical properties of the estimated random variate, some techniques may be more suitable to a certain simulation problem than others. Based on the findings of Kemna and Vorst [9], the method of control variates looks the most promising in the case of Asian option valuation. We discuss this in more detail in the next section, where we look into the properties of different simulation techniques, and explain how they could be efficiently implemented in option pricing.

2.3. Pricing Asian Options

After this brief introduction to the valuation framework, let us introduce a numerical valuation method based on the work of Geman and Yor [3], then we describe the simulation methods used later in Section 3. In the case of arithmetic-average Asian options, which we from now on refer to as Asian options for the sake of simplicity, the payout in Equation (1) is determined by the arithmetic mean of the underlying product's prices. In the continuous case

$$Avg_{[0,T]}\left(S^{(r)}\right)=\frac{1}{T}\int_0^T S^{(r)}(t)\mathrm{d}t \triangleq \frac{A(T)}{T}, \tag{6}$$

where $S^{(r)}$ is the risk-neutral underlying product process defined earlier. All we need to do now is apply the risk neutral valuation principle, as described in Equation (5), and calculate the call option price:

$$C=\mathrm{e}^{-rT}E_{\mathbb{P}}\left[\left(Avg_{[0,T]}\left(S^{(r)}\right)-K\right)^{+}\right]=\mathrm{e}^{-rT}E_{\mathbb{P}}\left[\left(\frac{A(T)}{T}-K\right)^{+}\right]. \tag{7}$$

Once the call option price is calculated, we may use the put-call parity, which is valid for all European-exercise options, to arrive at the put option price.

Lemma 1 (Put-Call Parity)

$$\begin{aligned} P&=\mathrm{e}^{-rT}E_{\mathbb{P}}\left(K-\frac{A(T)}{T}\right)^{+}\\ &=\mathrm{e}^{-rT}\left[E_{\mathbb{P}}\left(\frac{A(T)}{T}-K\right)^{+}-E_{\mathbb{P}}\left(\frac{A(T)}{T}-K\right)\right]\\ &=C-F+\mathrm{e}^{-rT}K \end{aligned} \tag{8}$$

where F is the "Asian forward price", as defined in the next lemma.

Lemma 2 (Asian Forward Price) *The forward price of the arithmetic-average of*

stock prices for a given time interval $[0,T]$ *is given by*

$$F \triangleq \mathrm{e}^{-rT} E_{\mathbb{P}}\left(\frac{A(T)}{T}\right) = \mathrm{e}^{-rT}\frac{S(0)}{T}\frac{4}{\sigma^2}\cdot\frac{\mathrm{e}^{2h(\nu+1)}-1}{2(\nu+1)}, \tag{9}$$

where

$$\nu \triangleq \left(r-\frac{\sigma^2}{2}\right)\frac{2}{\sigma^2} = \frac{2r}{\sigma^2}-1.$$

Therefore, in theory we are able to price Asian options. However, solving the simple-looking pricing formula in Equation (7) hides several pitfalls haunting financial mathematicians for decades. The fundamental problem is caused by the unknown probability distribution of $A(T)$. More specifically, the sum of lognormally distributed random variates is not lognormal, which makes the expected value in Equation (7) hard to calculate. Even though based on a result of Mitchell [10], who show that the sum of lognormals is close to lognormal, Turnbull *et al.* [2] derive an approximation for the option value, this could only be considered as a control tool rather than a sound pricing formula by market practitioners and risk managers.

2.3.1. The Laplace Transform Method

The breakthrough can be credited to Geman and Yor [3], based on whose seminal paper a series of authors, including Carr and Schröder [11] as well as Dufresne [12], conduct an exciting discussion over the validity and applicability of their approach. Before stating the main result, let us do a crucial transformation to the expression in Equation (7):

$$\begin{aligned} C &= \mathrm{e}^{-rT} E_{\mathbb{P}}\left(\frac{A(T)}{T}-K\right)^{+} \\ &= \mathrm{e}^{-rT}\cdot E_{\mathbb{P}}\left(\frac{S(0)}{T}\frac{4}{\sigma^2}A^{(\nu)}\left(\sigma^2T/4\right)-K\right)^{+} \\ &= \mathrm{e}^{-rT}\frac{S(0)}{T}\frac{4}{\sigma^2}\cdot C^{(\nu)}\left(\frac{\sigma^2T}{4},\frac{KT\sigma^2}{4S(0)}\right), \end{aligned} \tag{10}$$

where

$$A^{(\nu)}(T) \triangleq \int_0^T \exp\left\{2\left(\nu\cdot t + W_{\mathbb{P}}(t)\right)\right\}\mathrm{d}t \tag{11}$$

$$C^{(\nu)}(T,K) \triangleq E_{\mathbb{P}}\left(A^{(\nu)}(T)-K\right)^{+}, \tag{12}$$

the latter being referred to as the "normalized price" of the Asian option. Following [11], in the rest of this paper let us denote

$$h \triangleq \frac{\sigma^2T}{4} \quad \text{and} \quad q \triangleq \frac{KT\sigma^2}{4S(0)}.$$

The revolutionary finding of Geman and Yor [3] is that there exists an analytical formula for the Laplace transform of $C^{(\nu)}(h,q)$ with respect to h.

Theorem 2.6 (Geman-Yor) *The Laplace transform of the Asian option's normalized price, defined in Equation* (12), *with respect to h is*

$$\begin{aligned}\mathcal{L}\left\{C^{(\nu)}(h,q)\right\}(\lambda,q,\nu) &= \int_0^\infty \mathrm{e}^{-\lambda h} C^{(\nu)}(h,q)\,\mathrm{d}h \\ &= \frac{(2q)^{1-\beta}}{2\lambda(\alpha+1)\Gamma(\beta)} \int_0^1 \exp\left\{-\frac{u}{2q}\right\} u^{\beta-2}(1-u)^{\alpha+1}\,\mathrm{d}u,\end{aligned} \tag{13}$$

where

$$\gamma \triangleq \sqrt{2\lambda+\nu^2}, \quad \alpha \triangleq \frac{\gamma+\nu}{2}, \quad \beta \triangleq \frac{\gamma-\nu}{2},$$

for every

$$\lambda > \max\{0, 2(\nu+1)\}.$$

Using this result, it is possible to determine the option's price by first calculating the above Laplace transform, then inverting it to yield the normalized option price and finally substituting into Equation (10). However, as described in Cohen [13], the problem of inverting the Laplace transform is a nontrivial one (apart from a few special cases), and one often has to resort to numerical inversion techniques[2]. There have been various techniques invented, based on either of the following two inversion theorems.

Theorem 2.7 (Theorem) *If the Laplace transform* $\mathcal{L}$ *of a continuously differentiable function f exists, then*

$$f(t) = \mathcal{L}^{-1}\{\mathcal{L}(\lambda)\}(t) \triangleq \frac{1}{i(2\pi)} \cdot \lim_{T\to\infty} \int_{\gamma-iT}^{\gamma+iT} \mathrm{e}^{\lambda t}\mathcal{L}(\lambda)\,\mathrm{d}\lambda, \tag{14}$$

where γ is a real number so that $\mathcal{L}$ has no singularities on or right from γ on the complex plane.

Theorem 2.8 (Post-Widder Theorem) *If the Laplace transform* $\mathcal{L}$ *of a continuously differentiable function f exists, then*

$$f(t) = \mathcal{L}^{-1}\{\mathcal{L}(\lambda)\}(t) \triangleq \lim_{n\to\infty} \frac{(-1)^n}{n!}\left(\frac{n}{t}\right)\mathcal{L}^{(n)}\left(\frac{n}{t}\right),$$

where $\mathcal{L}^{(n)}$ is the n-th derivative of the particular Laplace transform.

Although in contrast to the Bromwich formula the Post-Widder formula is restrained to the real line and thus looks easier to handle, its speed of convergence is notoriously slow and the techniques based on them are prone to computational difficulties, as we can see in Fu *et al.* [5]. Hence, the two algorithms that we use are based on the Bromwich theorem, serving as a check for each other. In what follows, we present these algorithms described in Abate and Whitt [14].

The Euler Algorithm

Abate and Whitt [15] call this algorithm the Euler algorithm, since they employ Euler summation to accelerate convergence. Without going into further details, their basic idea is to look at the Bromwich integral in Equation (14), as though it was a Fouri-

[2]As the Laplace transform is unique, in the sense that if $\mathcal{L}\{f\} = \mathcal{L}\{g\}$, then $f = g$, we need not worry about one approach leading to a different result than another.

er-series, by substituting for $\lambda = a + iu$:

$$\begin{aligned}\mathcal{L}^{-1}\{\mathcal{L}(\lambda)\}(t) &= \frac{1}{i(2\pi)} \cdot \lim_{T\to\infty} \int_{\gamma-iT}^{\gamma+iT} \mathrm{e}^{\lambda t}\mathcal{L}(\lambda)\mathrm{d}\lambda \\ &= \frac{1}{2\pi} \cdot \lim_{T\to\infty} \int_{-T}^{T} \mathrm{e}^{(\gamma+iu)t}\mathcal{L}(\gamma+iu)\mathrm{d}u \\ &= \frac{\mathrm{e}^{\gamma t}}{2\pi} \int_{-\infty}^{\infty} \mathrm{e}^{iut}\mathcal{L}(\gamma+iu)\mathrm{d}u.\end{aligned}$$

Their final result is the algorithm which also Equation [5] used in their calculations, being apparently a very (if not the most) efficient numerical algorithm in inverting the Laplace transform derived by Geman and Yor [3].

Algorithm 2.9 (Euler) *Let us define*

$$s_n(t) \doteq \frac{\mathrm{e}^{A/2}}{2t} Re\left\{\mathcal{L}\left(\frac{A}{2t}\right)\right\} + \frac{\mathrm{e}^{A/2}}{t}\sum_{k=1}^{n}(-1)^k a_k(t)$$

and

$$a_k(t) \doteq Re\left\{\mathcal{L}\left(\frac{A+i(2\pi)k}{2t}\right)\right\}.$$

Then, the inverse Laplace transform could be approximated by the Euler series

$$\mathcal{L}^{-1}\{\mathcal{L}(\lambda)\}(t) \approx \hat{f}(t,A,m,n) = \sum_{k=0}^{m}\binom{m}{k}2^{-m}s_{n+k}(t),$$

where A, m and n are chosen parameters of the algorithm, for which Abate and Whitt [15] recommend $A = 18.4$, $m = 11$ and $n = 15$ (increasing n as necessary).

The Talbot Algorithm

Similarly to the Euler algorithm, this algorithm is also based on the Bromwich theorem. The idea of Talbot, described in Cohen [13], is to mitigate the disturbing effects of oscillation as we move along the integration contour parallel to the imaginary axis by deforming (*i.e.* altering) the integration contour to a more appropriate one (let us call it Γ), which encloses all singularities of $\mathcal{L}$ in the complex plane. This idea can be formally summarized as

$$\mathcal{L}^{-1}\{\mathcal{L}(\lambda)\}(t) = \frac{1}{i(2\pi)} \cdot \lim_{T\to\infty} \int_{\gamma-iT}^{\gamma+iT} \mathrm{e}^{\lambda t}\mathcal{L}(\lambda)\mathrm{d}\lambda = \frac{c\mathrm{e}^{\sigma t}}{i(2\pi)} \int_{\Gamma} \mathrm{e}^{c\lambda t}\mathcal{L}(\sigma+c\lambda)\mathrm{d}\lambda,$$

which is applied by Abate and Whitt [14] in the following numerical inversion algorithm.

Algorithm 2.10 (Talbot) *Let us define*

$$\delta_0 \doteq \frac{2M}{5}, \qquad \delta_k \doteq \frac{(2\pi)k}{5}\left(\cot(k\pi/M)+i\right)$$

and

$$\gamma \doteq \frac{1}{2}\mathrm{e}^{\delta_0}, \qquad \gamma_k \doteq \left[1+i(k\pi/M)\left(1+\cot^2(k\pi/M)\right)-i\cot(k\pi/M)\right]\mathrm{e}^{\delta_k}.$$

Then, the inverse Laplace transform could be approximated by the series

$$\mathcal{L}^{-1}\{\mathcal{L}(\lambda)\}(t) \approx \hat{f}(t,M) = \frac{2}{5t}\sum_{k=0}^{M-1} Re\left\{\gamma_k \mathcal{L}\left(\frac{\delta_k}{t}\right)\right\},$$

where M is a chosen parameter of the algorithm.

Implementation

By means of the above algorithms, the Laplace transform

$$\mathcal{L}(\lambda) \equiv \mathcal{L}\left\{C^{(\nu)}\right\}(\lambda, q, \nu)$$

could now be numerically inverted with respect to λ to obtain

$$f(h) \equiv C^{(\nu)}(h,q).$$

We use MATLAB® for the implementation of the introduced algorithms. The codes are presented in Sections A.1 and A.2 of the Appendix. For the sake of clarity, we strive to preserve the notation presented earlier in this paper and insert as many comments into the code as possible. Without going into unnecessary technicalities, let us point out a couple of practical features and enhancements in the code:

- Both algorithms are based on substituting back into the Laplace transform $\mathcal{L}$ at different loci on the complex plane. Most MATLAB® functions can take complex arguments, so this should not be a setback. One important hint to bear in mind though is the use of "*1i*" instead of "*i*" in the code, indicating that we refer to the imaginary number and not a variable.
- A crucial step in calculating the Laplace transform presented in Equation (13) is complex integration on the $(0,1)$ interval. Since solving this problem is not in the scope of our research, we use the built-in "integral" function for this purpose, which is in turn based on the proprietary "global adaptive quadrature" of MathWorks®. In addition, at this point we feel obliged to emphasize the considerable achievement of Geman and Yor [3] in providing an integral on a closed interval, as the handling of improper integrals would have been a numerically much less tractable problem.
- Based on our experience, during the implementation of the algorithms we have to work with quantities of very different magnitudes. More specifically, the integral in Equation (13) is usually of magnitude 10^{-500} to 10^{-100}, the preceding fraction being large enough to compensate. This poses a big challenge because computers use floating-point numbers, and thus multiplication and division of such different magnitude numbers could lead to the complete loss of precision. The measures taken to mitigate this problem are

1) Taking the (natural) logarithm of the whole expression in Equation (13) so as to exponentially reduce the magnitude of variables. Although this step does not necessarily increase precision, it definitely "earns space" in terms of magnitude.

2) Explicitly instructing MATLAB® to use the Symbolic Math Toolbox®, which allows the user to choose an arbitrary level of precision for calculations. With regard to our particular problem, we consider a precision of 25 significant digits sufficient.

Having understood the numerical approach based on the Laplace transform of the Asian option's normalized price, we are ready to price Asian options with two algo-

rithms. But before proceeding to the empirical analysis concerning the precision and robustness of these algorithms, let us review the simulation approach, which will serve as an ultimate check and basis of comparison for the Laplace transform method.

2.3.2. The Monte Carlo Method

Keeping in mind the basics of the Monte Carlo method from the end of Section 2.2, we know that this is a highly-flexible, robust approach, suitable to various frameworks and product types. Flexibility, however, has a price taking the form of variance in the case of simulation methods, which means that the simulated price bears some uncertainty and fluctuates around the option's true (analytical) value. Striving to cut this variance to a minimum level, we study in details how the variance reduction techniques of antithetic variates and control variates work in the case of financial derivative valuation.

Basic Method

As the name suggests, this is the simplest form of Monte Carlo method, taking the arithmetic-mean as an unbiased estimator of the expected value:

$$\hat{E}(X) \stackrel{\circ}{=} \frac{\sum_{i=1}^{n} X_i}{n} = \sum_{i=1}^{n} \frac{X_i}{n},$$

which has a variance of

$$D^2\left(\hat{E}(X)\right) = D^2\left(\sum_{i=1}^{n} \frac{X_i}{n}\right) = \frac{D^2(X)}{n},$$

where $X_i = f(\omega_i)$ are IID random variates representing the ith outcome in our simulation. The random function f, whose expected value we wish to estimate is the discounted payout function of the option as suggested by the principle in Equation (5):

$$f(\omega) \stackrel{\circ}{=} \mathrm{e}^{-rT} \times Payout(\omega),$$

which, in the particular case of Asian call options, takes the form of

$$f(\omega) = \mathrm{e}^{-rT}\left(Avg_{[0,T]}\left(S^{(r)}(\omega)\right) - K\right)^{+},$$

having an expected value

$$E\left(f(\omega)\right) = \mathrm{e}^{-rT} E_{\mathbb{P}}\left[\left(Avg_{[0,T]}\left(S^{(r)}(\omega)\right) - K\right)^{+}\right],$$

which is precisely the price of the option given in Equation (7). As we can see, it is this risk-neutral expected value that we estimate through simulation of the underlying product's risk-neutral trajectory

$$\left\{S_t^{(r)}(\omega)\right\}_{t\in[0,T]},$$

which will be constructed as

$$\left\{S_{t_j}^{(r)}(\omega)\right\}_{j\in[0,m\cdot(T\times 365)]} \stackrel{\circ}{=} \left\{S(0)\cdot\exp\left\{\sum_{i=0}^{j} X_{t_i}(\omega)\right\}\right\}_{j\in[0,m\cdot(T\times 365)]},$$

where

$$X_{t_i}(\omega) \triangleq \begin{cases} 0 & \text{if } i = 0 \\ \left(r - \dfrac{\sigma^2}{2}\right)(t_i - t_{i-1}) + \sigma\left(W(t_i) - W(t_{i-1})\right) & \text{if } i > 0 \end{cases}$$

is the risk-neutral log-return applying Equation (2) and m indicates the number of daily reads (*i.e.* the mesh of the equidistant time grid).

We must point out that in the case of path-dependent derivatives simulation methods have two sources of estimation error:

1) The generated trajectories (outcomes) depend on the random event ω, and thus the calculated estimator is also a random variate, fluctuating around the expected value that we look for. This *simulation error* could be reduced by increasing *n*, which is the number of trajectories (outcomes) in our simulation.

2) The generated trajectories are discretized, which means they are only approximations of the true, continuous-time trajectories required for the calculation of the path-dependent derivative payout. In other words, we substitute the integral with a sum in the payout of the Asian option and no matter how great *n* and thus how accurate our estimator is, we still face some discretization error. This error could be reduced by increasing m, which is the number of fragments we break up each day of interval $[0,T]$ into.

Method of Antithetic Variates

Keeping the logic and weak points of the basic simulation method in mind, let us see how we could improve on our estimate of the expected value. First of all, let us try and reduce the variance of the estimator. A suitable technique is the method of antithetic variates, described in Kleijnen *et al.* [8]. The principal idea here is using each generated random number twice, creating two trajectories from each log-return sequence, where one is negatively correlated (antithetic) to the other. Although the way the antithetic trajectories are created is greatly product specific, it is usually a good idea to find a step in the simulation algorithm when we deal with symmetrically distributed random variates, and then simply "mirror" them about their expected value. In our particular case of log-return sequences, we simply take the negative of each log-return and thus create a "mirrored trajectory". Calling them sample A and sample B, let us create a new estimator

$$\hat{E}_{anti}(X) \triangleq \frac{\hat{E}(X_A) + \hat{E}(X_B)}{2} = \sum_{i=1}^{n/2} \frac{X_i + X_i^{(a)}}{n},$$

where X_i and $X_i^{(a)}$ are the payouts based on the i[th] original and antithetic trajectory, respectively. This estimator has a variance of

$$\begin{aligned} D^2\left(\hat{E}_{anti}(X)\right) &= D^2\left(\sum_{i=1}^{n/2} \frac{X_i + X_i^{(a)}}{n}\right) \\ &= \frac{D^2(X)/2 + D^2\left(X^{(a)}\right)/2 + Cov\left(X, X^{(a)}\right)}{n} \\ &\approx \frac{D^2(X) + Cov\left(X, X^{(a)}\right)}{n}, \end{aligned} \tag{15}$$

where we use the symmetric nature of the random walk. The benefit of using this method is twofold:

1) It is apparent from the above results that as long as the original and antithetic payouts have a negative covariance, the variance of the estimator will be reduced compared to that of the basic method, while the estimator remains unbiased. As we demonstrate in Section 3, thanks to the low correlation of the original and the mirrored trajectories, the variance reduction achieved by this relatively simple method is substantial.

2) Since we use every random number twice, we only need to generate $n/2$ outcomes instead of the original n ones, which means that the method of antithetic variates is (almost) twice as efficient as the basic method.

Method of Control Variates

As demonstrated through the above example of antithetic variates, it is possible to keep the estimator unbiased, while significantly reducing its variance. As it turns out, it is possible to further improve on our simulation technique by using control variates instead of antithetic variates (for further details refer to Kleijnen *et al.* [8]). This method is based on the logic that if we add an appropriate (*i.e.* highly correlated, zero expectation) expression to our original estimator, then variance reduction could be achieved while the estimator remains unbiased. Let us denote the control variate with Y, while preserving our original variate X. In this case, the estimator is

$$\hat{E}_{control}(X) \doteq \hat{E}(X) + c\hat{E}(Y) = \sum_{i=1}^{n} \frac{X_i + cY_i}{n},$$

where c is a properly chosen[3] constant, X_i and Y_i are the i[th] outcomes of the original and control variates in our simulation, respectively. It is easy to see that the variance will be

$$\begin{aligned} D^2\left(\hat{E}_{control}(X)\right) &= D^2\left(\sum_{i=1}^{n} \frac{X_i + cY_i}{n}\right) \\ &= \frac{D^2(X) + c^2 D^2(Y) + 2cCov(X,Y)}{n}. \end{aligned} \tag{16}$$

Kemna and Vorst [9] show that in the case of the arithmetic-average Asian option price, the use of the geometric-average Asian option price can be used as an effective control variate. In a succeeding research, Fu and Madan [5] compare the discrete and the continuous geometric-average Asian option price's performance as control variate with interesting results. They find that even though the appropriate formula for a discrete estimator would be the discrete average formula, by using the continuous average formula a greater variance reduction can be achieved. In other words, by using a "biased" (*i.e.* non-zero expectation) control variate, we not only reduce the variance of our original estimator, but also add an appropriate bias to it, which compensates for the above mentioned discretization error inherent in the basic simulation method. For this

[3]It can be easily shown that the optimal value of the constant is $c^* = -\frac{Cov(X,Y)}{D^2(Y)}$, which minimizes the variance of the estimator.

reason, we also use this "discretization adjusted" control variate in our simulations. It is crucial to note the advantages of this method:

1) It reduces the simulation error of our original estimator introduced in the basic method, even more so than the method of antithetic variates[4].

2) It effectively eliminates some of the discretization error inherent in the simulation technique by using a "discretization adjusted" control variate.

2.3.3. Implementation

Before proceeding with the empirical analysis of our valuation framework in the next section, let us make a few notes with regard to the practical implementation of the simulation methods discussed above. Firstly, we perform the simulation following the algorithms presented in this section. Although the true variances and covariances are unknown, they could be estimated from the sample. Secondly, the greatest challenge during the simulation is to avoid running out of memory. Thanks to the capacity of modern computers, this is usually not an issue when simulating only the endpoints of trajectories as in the case of European-exercise vanilla options. However, when the entire trajectory needs to be generated and stored in the memory (e.g. in the case of path-dependent derivatives), we run out of resources alarmingly fast. Without going into further technical details (e.g. issues with "memory paging"), what we do to avoid this eventuality is break each trajectory into small enough sections that the generated log-return matrix fits into the memory. Then, we roll over the trajectory, calculating the (geometric) mean step by step, efficiently reusing the available memory space again and again as shown in **Figure A5** Memory of the Appendix.

3. Empirical Analysis

After the introduction of the different approaches to Asian option pricing, now we take a more practical perspective and compare the efficiency and computational characteristics of the methods described in Section 2.3 by analyzing outputs from MATLAB®. The computations are performed under Windows 7 operation system by means of an Intel® Core i7 2.2 GHz CPU 8 GB RAM hardware.

First, we examine how the two semi-analytical (*i.e.* Euler and Talbot) algorithms fare compared to each other, then we contrast simulation outputs for different reading frequencies and sample sizes. Then, we compare the semi-analytical and the simulation approaches and make suggestions on their appropriate usage in real life situations.

3.1. Numerical Results

At first glance the semi-analytical algorithms presented in the previous section might look awkward but in effect they are very efficient for certain parameter ranges of the Asian call option price $C(S,K,r,T,\sigma)$. As Fu and Madan [5] point out, "difficulties [...] seem to begin when $\sigma^2(T-t)<0.01$". Our calculations confirm that the critical

[4]Based on our experience, the correlation between the arithmetic-average Asian option price and the geometric-average Asian option price is very high (above 0.9), which indicates that a very significant variance reduction could be achieved. For further details, refer to Section 3.

parameter is indeed $\sigma^2 T$, which fundamentally governs how far away the underlying product's price can get from its current state during the lifetime of the option, and directly determines the locus h at which the normalized option price $C^{(\nu)}(h,q)$, defined in Equation (12), must be calculated.

For the sake of simplicity, let us fix σ at one[5] and observe what effect changes in T (*i.e.* time to maturity) have on an ATM ($S=K=1$) Asian call option price's calculation efficiency. As hinted in Section 2.3.1, the main parameters of the Euler and Talbot algorithms, driving computational precision, are n and M, respectively. Leaving the other parameters at their default values[6] as suggested by Abate and Whitt [15], the way we measure the efficiency of each algorithm is iteratively increasing the precision parameters until the last four decimal digits of the option price is stabilized. The results are shown in **Figure 1**, where we depict in log-log charts the times and parameter levels that each algorithm required to reach the desired precision. From these charts, we may conclude the following:

- The Euler algorithm is more robust than the Talbot algorithm with respect to both calculation time and parameter levels. In general, setting $n=10$ produces an Asian option value of at least four significant digit precision.
- The Talbot algorithm is more reliable than the Euler algorithm in the sense that if m is set sufficiently high, then the precise Asian option value can be calculated even for small values of $\sigma^2 T$ (given enough computation time). In contrast, the Euler algorithm fails to converge when $\sigma^2 T < 0.005$, no matter how high n is set.
- When $\sigma^2 T$ falls below a sufficiently low threshold, we can see a rapid increase in both calculation times and parameter levels. However, when $\sigma^2 T$ remains outside the "danger zone", the semi-analytical method proves to be extremely efficient and solve the inversion problem under 2 - 3 seconds.

By repeating the above calculation test with different combination of option parameters, we conclude that it is only the value of $\sigma^2 T$ which determines the speed of the

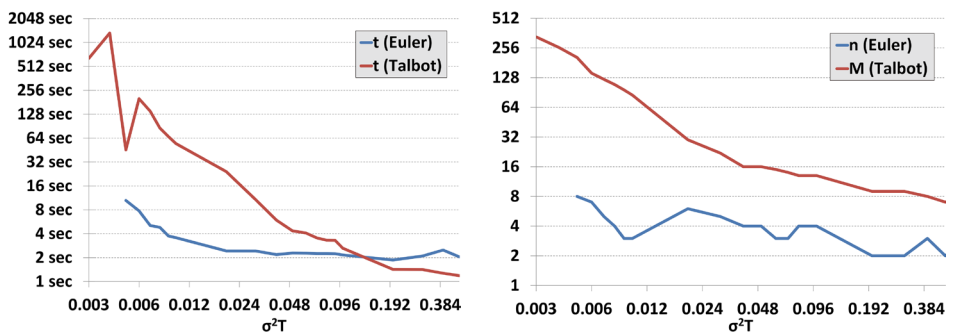


Figure 1. Comparison of the Euler and Talbot analytical algorithms.

[5]Naturally, $\sigma=1$ is an unrealistic scenario, but since it is the value of $\sigma^2 T$ that drives the Asian option price given in Equation (10), it is a convenient choice, making it easier for us to capture the effects of changes in T.

[6]Fu *et al.* [5] does an extensive research on tweaking the Euler algorithm's other two parameters (*i.e.* A and m), concluding that the default setting (*i.e.* $A=18.4$ and $m=11$) suggested by Abate and Whitt [15] is indeed optimal.

Laplace transform inversion approach. An important question to be answered though is why the chosen numerical algorithms have convergence issues when $\sigma^2 T$ is small. Considering that this is a phenomenon that also other authors, such as Fu *et al.* [5] and Craddock *et al.* [16] confirm, it seems sensible to assume that as $\sigma^2 T \to 0$, inversion complications arise due to the inherent numerical characteristics of the Laplace transform calculated by Geman and Yor [3], no matter which inversion algorithm we choose. Indeed, observing the charts presented in Section A.3 of the Appendix, we find that as $\sigma^2 T \to 0$, the Laplace transform gets "numerically out of hand":

- In the case of the *Euler algorithm*, the Laplace transform oscillates and quickly tends to zero as k (*i.e.* the locus in the complex plane) changes, which limits the effect of the precision parameter, there being no reason to use $n > 20$ in the calculations. The problem is that as $\sigma^2 T \to 0$, the Laplace transform rapidly diminishes to orders smaller than 10^{-10}, making the embedded numerical integration, which is outlined in Equation (13), increasingly hard to calculate with the desired precision.
- In the case of the *Talbot algorithm*, the Laplace transform shows some oscillating nature again as k (*i.e.* the locus in the complex plane) changes but remains zero outside a given range. The problem here is not that the transform diminishes to magnitudes which are hard to handle numerically but that this specific location of oscillation is unknown, moreover, it shifts along the k axis towards infinity as $\sigma^2 T$ gets smaller. It follows that in order to reach the desired precision, we have to set k large enough that this location of interest is included in the final summation within the algorithm. This entails the usage of an ever larger summation, or in other words the value of precision parameter M (and thus the calculation time) needs to be increased as $\sigma^2 T \to 0$.

Although with the Euler algorithm we quickly run into a "dead-end street", since the embedded integral cannot be calculated numerically to an arbitrary precision, the deformation of the Bromwich-contour in case of the Talbot algorithm seems to leave some space for improvement. If we could (either numerically or analytically) pinpoint the location of oscillation without having to calculate all the zeros before, we could significantly improve on the algorithm's performance.

Moreover, we consider it an achievement that we could significantly reduce calculation times compared to the ones produced by Fu *et al.* [5] as shown in **Figure A6** in the Appendix. The former calculation time of 30 - 40 seconds have been reduced to a mere 3 - 4 seconds. How much of this achievement could be attributed to better implementation and how much to better equipment, though, is hard to judge.

3.2. Simulation Results

In the previous section we investigate how the two numerical algorithms perform for different combinations of option parameters. Before making a comparison with the simulation approach, let us look into the latter on its own and examine the three Monte Carlo algorithms introduced in Section 2.3.2. As explained earlier, in the case of path-dependent derivatives (such as Asian options) simulation techniques have two govern-

ing parameters:

- n, which is the number of trajectories simulated,
- m, which is the number of fragments each day is partitioned into,

where n determines the simulation error, whereas m determines the discretization error. Striving to isolate the effects of these parameters as much as possible, first we examine what level of m leads to a satisfactory approximation of the continuous Asian option price, and only then will we turn to adjusting (lowering) n in order to attain an estimate with desirable precision. Just as in the case of numerical methods, we constrain our analysis to ATM ($S = K = 1$) Asian call options and $\sigma = 1$. The benefit of this restriction is twofold. On the one hand, simulation results will be easily comparable to the respective numerical results. On the other hand, the relationship between T and the estimation error will serve as a good proxy for the relationship between $\sigma^2 T$ and the estimation error.

However, before proceeding with the fine-tuning of the above parameters and measuring the performance of the different simulation algorithms, let us have a word on what level of error we consider satisfactory. In order to make a later comparison between simulation and numerical methods possible, we aim at the same (*i.e.* four-decimal) valuation precision. To achieve this, we need to set the simulation parameters so that the confidence interval[7] around the estimated option value is sufficiently small to ensure the desired precision. Since the exact probability distribution of the Asian option price is unknown, let us suffice with Chebyshev's inequality and give an approximation for the confidence interval by

$$P\left(\left|E(X) - X\right| \geq k\sigma\right) \leq \frac{1}{k^2},$$

which implies that by choosing $k = 4$, we can be about 95% certain that the true Asian option value falls between $\hat{E}(X) - 4\hat{\sigma}$ and $\hat{E}(X) + 4\hat{\sigma}$. In other words, the estimated standard deviation has to be of order 10^{-5} so that the desired four-decimal precision is satisfied.

3.2.1. Discretization Error

Keeping in mind the findings about simulation precision, let us determine what level of m ensures a good approximation of the underlying product's continuous trajectory. If we set $n = 100000$, which is relatively large, we expect to receive results with low simulation error. It follows that by changing m, we should be able to discern changes in discretization error, only slightly disturbed by any residual simulation error in the output. Applying this logic, we run the simulation for different maturities and gathered the output in Section A.4 of the Appendix, displaying the absolute error (*i.e.* the absolute difference of the simulated option value and the true option value). From these charts, we can make the following observations:

- The control variate method has lower (discretization) error than the other two methods by about one order of magnitude.

[7]In order to define a confidence interval, we should also pick a suitable significance level. Let this be 95%.

- There is a rapid decrease in discretization error when using $m=10$ instead of $m=1$. However, the discretization error reduction is not significant when m is further increased to $m=100$.
- Interestingly, the antithetic variate method does not reduce but rather amplifies *discretization* error. On a second thought, this phenomenon is not surprising at all, as in the case of the antithetic variate method each generated random number is used twice, so as to cut down on simulation error and time. However, this method also entails that any discretization error in a given simulated trajectory is emphasized (duplicated) as there are less independent trajectories used in the simulation process.
- By taking a closer look at the absolute errors for different levels of m, we might notice patterns, such as peaks or troughs, especially in case of the first two methods. This is not a *coincidence*, as we use the same random number sequences for each valuation, which means that longer simulations (*i.e.* where T is greater) use the same random numbers as their shorter counterparts, and thus any discretization error will also be preserved. This "slowly decaying" nature of the estimation error is very useful, making it easier for us to identify those simulation methods which effectively reduce discretization error. For instance, it is very hard to discern patterns in the chart of the control variate method, which indicates that this method does exactly what it professes, namely that it reduces discretization error inherent in the simulation of the arithmetic-average by offsetting it with the one inherent in the simulation of the geometric-average.

From the above results, we can conclude that it is best to choose the control variate method, as it effectively reduces discretization error. Also, it is unnecessary to set m greater than 10, as creating more refined trajectories significantly increases simulation time but does not attain a proportionate discretization error reduction. Hence, in the rest of our work we set $m=10$.

3.2.2. Simulation Error

From a perspective of discretization error reduction, the method of control variates is clearly the best performing one. Now, let us make a comparison between simulation methods based on the simulation error of their outputs, measured by the standard deviation of the estimate.

The estimated standard errors produced by each simulation method (using a given number of trajectories) are shown on a log-log scale in **Figure 2**, from which we draw the following inferences:

- The antithetic method achieves only a slight (about 15% - 20%) reduction in simulation variance compared to the basic method.
- The control variate method achieves a substantial (about 90% - 95%) reduction in simulation variance compared to the basic method. The variance reduction is so efficient that as few as $n=10000$ trajectories could be sufficient to make an accurate estimate (especially when $\sigma^2 T$ is relatively small).

- The standard error from regular and antithetic methods grows only at a rate $\sim x^{0.55}$, whereas the control variate method's standard error grows at a faster rate $\sim x^{1.05}$. It looks as though the significant discretization error reduction observed earlier has a price, and hence the standard error of the control variate method catches up with the standard error of other methods as $\sigma^2 T$ increases.

The question naturally arises whether we quantify the variance reduction by the antithetic and the control variate methods. For this, we need to estimate the correlation of antithetic variates and control variates, respectively. As we can see in Figure 3, these correlations are not constants but they depend on the value of T (*i.e.* the length of the simulated trajectories).

We know from Equations (15) and (16) that

$$D^2\left(\hat{E}_{anti}(X)\right) \approx \frac{D^2(X) + Cov\left(X, X^{(a)}\right)}{n} \approx \frac{D^2(X)\left(1+\rho\left(X, X^{(a)}\right)\right)}{n}$$

and (for a variance-optimal c^*)

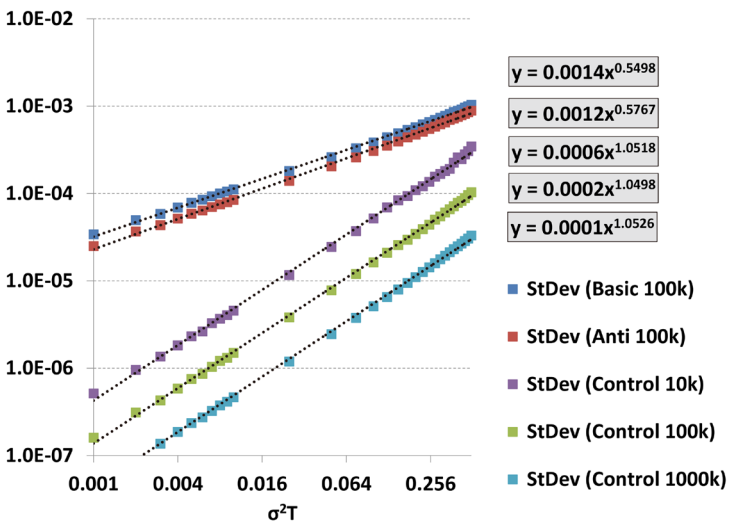


Figure 2. Comparison of the different simulation methods ($m = 10$).

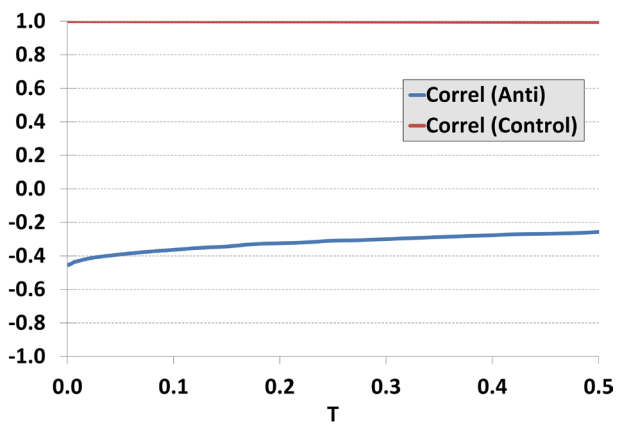


Figure 3. Correlation of proxy variates at variance reduction methods.

$$D^2\left(\hat{E}_{control}(X)\right)=\frac{D^2(X)+c^2D^2(Y)+2cCov(X,Y)}{n}$$
$$\approx\frac{D^2(X)-\frac{Cov^2(X,Y)}{D^2(Y)}}{n}\approx\frac{D^2(X)\left(1-\rho^2(X,Y)\right)}{n},$$

provided $D^2(X)\approx D^2\left(X^{(a)}\right)$ and $D^2(X)\approx D^2(Y)$. Comparing these simulation errors with the variance of the basic method, which is

$$D^2\left(\hat{E}(X)\right)=\frac{D^2(X)}{n},$$

we can quantify the variance reduction. In the case of the antithetic method, $\rho\in(-0.4,-0.2)$ ensures a 20% - 40% variance reduction, which is indeed a 10% - 20% standard error reduction. In contrast, $\rho\approx 1$ in the case of the control variate method results in a close to 100% standard error reduction, just as experienced earlier. Also, by looking at the above calculations, where $D^2(X)$ is multiplied by $(1+\rho)$ in the antithetic and by $\left(1-\rho^2\right)$ in the control variate case, we understand why the simulation error of the latter grows faster as $\rho\to 0$.

From the results presented we may conclude that the control variate method effectively reduces both discretization error and simulation error even for $m=10$. Using this method, the estimation error remains below the desired 10^{-5} if n is appropriately chosen as per Table 1.

3.3. Comparison of Valuation Approaches

Having compared the different numeric and simulation methods separately and having identified the optimal parameter values for each, let us compare the two approaches in the search for an answer to our second research question. Namely, given today's computational capacity what method (algorithm) should we use for pricing Asian options?

We have seen that the Euler and Talbot algorithms are powerful tools for pricing Asian options, though being rather sensitive to low values of σ^2T. Also, we have concluded that the control variate method is able to determine the price of Asian options very accurately by efficiently reducing both discretization and simulation variance. The only question remaining is how fast these algorithms are, compared to one another. Could we set up a preference order, maybe even as a function of σ^2T? To answer this question, let us have a look at Figure 4, where we can see what time it takes for

- the Euler algorithm to calculate the Asian option price to the desired four-decimal precision;

Table 1. Optimal parametrization of the control variate method ($m=10$).

σ^2T	n
<0.025	10000
<0.05	100000
<0.1	1000000

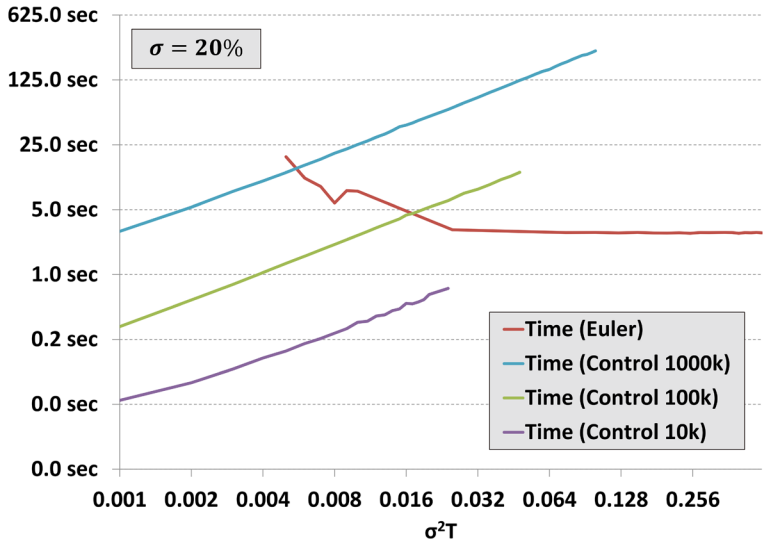


Figure 4. Comparison of calculation times using different methods ($m = 10$).

- the control variate algorithm to run a simulation for a given n, showing each curve only until the desired precision can be maintained.

Let us note that while the time required by numeric algorithms is fundamentally a function of $\sigma^2 T$, simulation times depend basically on T (*i.e.* how long trajectories need to be generated). This implies that for each σ, the relative position of timings changes, as we may observe the differences between **Figure 4** and **Figure A4** (in Section A.4 of the Appendix), calculated for $\sigma = 20\%$ and $\sigma = 10\%$, respectively[8].

Based on the figures, we claim that the control variate algorithm generally beats numerical algorithms in terms of calculation time for usual values of σ and T. However, we should also incorporate another factor in our analysis before making a conclusion, namely calculation precision. As we observe in **Figure 2** and **Table 1**, even when using the control variate method, the standard error of the estimate does not satisfy our desired level of precision when $\sigma^2 T$ exceeds a given limit, indicating that it might be worth to switch to numeric algorithms under such conditions. Based on our findings, we may set up the following decision rules to optimize the Asian option pricing methodology.

4. Conclusion

By maintaining a practical approach to the valuation problem in this paper, we pay special attention to the implementation of the numeric algorithms and to the specific computational issues that arose in MATLAB®. In Section 3, we test the two semi-analytical (numerical) methods and the three simulation methods for different option parameter settings, striving to answer the questions how the efficiency of the different algorithms as well as their relative performance have changed due to the technological development of the last decade.

[8]Under usual market conditions, the stock price volatility lies between these two values, and thus **Figure 4** and **Figure A4** show the extremities for the σ parameter.

Pointing out the importance of $\sigma^2 T$ in the Asian option pricing problem in Section 3.1, we manage to further improve on the reliability and efficiency of the numerical inversion algorithms, pushing the critical value of $\sigma^2 T$ down to 0.005 with 3 - 4 seconds calculation time (in comparison to the critical value of 0.01 with 30 - 50 seconds calculation time, as shown in the work of Fu *et al.* [5]). Also, based on the work of Abate and Whitt [14], we successfully employ the *Talbot algorithm as a viable alternative* to the Euler algorithm in Laplace transform inversion. Although the latter proved to be more robust, the Talbot algorithm showed superior computational efficiency (*i.e.* a valuation time of scarcely two seconds) for greater values of $\sigma^2 T$.

Based on the simulation results in Section 3.2 we confirm that the method of using the geometric-average Asian option price as control variate is an extremely efficient one, rendering traditional Monte Carlo methods useless in the particular valuation problem of arithmetic-average Asian options. Indeed, whereas for the latter methods the generation of 100 000 trajectories is not sufficient to price the option with the desired (four-digit) accuracy, the control variate method requires only 10,000 trajectories to fulfill our requirements, taking fractions of a second to compute the Asian option price.

Hence, we can experience an interesting dichotomy between numerical and simulation methods: one approach tends to excel when the other shows inferior performance and vice versa, depending on the value of $\sigma^2 T$ as shown in **Table 2**. It follows that the two approaches complement each other, so that there is always an applicable algorithm which is able to price an Asian option within 5 seconds with a precision of four decimal digits (in our perception, this speed and precision is acceptable under most market circumstances).

Regarding the comparison of the numerical and simulation approaches let us make two subtle but crucial remarks. Firstly, we cannot emphasize enough the geometric-average Asian option price's importance in this particular valuation problem. Had it not been for this control variate's fortunate availability, the simulation approach would have performed far worse than in this special case. Thus, before waving aside the Laplace transform inversion methods for their notorious sensitivity to small values of $\sigma^2 T$, we should remember that in valuation problems where there is no such control variate available these numerical algorithms could serve as a valuable pricing tool.

Secondly, let us point out the Laplace transform method's weakness, namely that it is limited by the assumptions of the Black-Scholes model. Presumably the strongest of these tenets is the constant volatility assumption, the relaxation of which could fundamentally alter the Laplace transform formula derived by Geman and Yor [3]. Also, if

Table 2. Optimal method selection rules for pricing Asian options.

$\sigma^2 T$	optimal method
<0.025	MC with control variate ($m=10$, $n=10k$)
<0.1	Euler algorithm ($n=10$)
>0.1	Talbot algorithm ($M=10$)

the assumption of geometric Brownian motion is relaxed (e.g. log-returns are modeled by the more general Lévy processes), the validity of the numerical results is questionable. In contrast, the flexibility of simulation methods makes it possible to adjust the original valuation framework to a wider range of assumptions, thus enabling market participants to use more realistic stock processes for Asian option pricing.

References

[1] Bodnar, G.M., Hayt, G.S. and Marston, R.C. (1998) Wharton Survey of Financial Risk Management by U.S. Non-Financial Firms. *Financial Management*, **27**, 70-91. http://dx.doi.org/10.21314/JCF.1998.024

[2] Turnbull, S.M., McLean, S. and Wakeman, L.M. (1991) A Quick Algorithm for Pricing European Average Options. *Journal of Financial and Quantitative Analysis*, **26**, 377-389. http://dx.doi.org/10.21314/JCF.1998.024

[3] Geman, H. and Yor, M. (1993) Asian Options, Bessel Processes and Perpetuities. *Mathematical Finance*, **3**, 349-375. http://dx.doi.org/10.21314/JCF.1998.024

[4] Geman, H. and Eydeland, A. (1995) Domino Effect: Inverting the Laplace Transform. *Risk*, **8**, 65-67.

[5] Fu, M.C., Madan, D.B. and Wang, T. (1999) Pricing Continuous Asian Options: A Comparison of Monte Carlo and Laplace Transform Inversion Methods. *Journal of Computational Finance*, **2**, 49-74. http://dx.doi.org/10.21314/JCF.1998.024

[6] Black, F. and Scholes, M. (1973) The Pricing of Options and Corporate Liabilities. *Journal of Political Economy*, **81**, 637-654. http://dx.doi.org/10.1086/260062

[7] Kalos, M.H. and Whitlock, P.A. (2008) Monte Carlo Methods. 2nd Edition, John Wiley & Sons, Inc., Hoboken. http://dx.doi.org/10.1002/9783527626212

[8] Kleijnen, J.P.C., Ridder, A.A.N. and Rubinstein, R.Y. (2010) Variance Reduction Techniques in Monte Carlo Methods. CentER Discussion Paper Series, No. 2010-117.

[9] Kemna, A.G.Z. and Vorst, A.C.F. (1990) A Pricing Method for Options Based on Average Asset Values. *Journal of Banking & Finance*, **14**, 113-129. http://dx.doi.org/10.1002/9783527626212

[10] Mitchell, R.L. (1968) Permanence of the Log-Normal Distribution. *Journal of the Optical Society of America*, **58**, 1267-1272. http://dx.doi.org/10.1364/JOSA.58.001267

[11] Carr, P. and Schröder, M. (2004) Bessel Processes, the Integral of Geometric Brownian Motion, and Asian Options. *Theory of Probability & Its Applications*, **48**, 400-425. http://dx.doi.org/10.1137/S0040585X97980543

[12] Dufresne, D. (2004) Bessel Processes and a Functional of Brownian Motion. Center for Actuarial Studies, Department of Economics, University of Melbourne, Parkville. http://hdl.handle.net/11343/34342

[13] Cohen, A.M. (2007) Numerical Methods for Laplace Transform Inversion. Springer Science & Business Media, Berlin.

[14] Abate, J. and Whitt, W. (2006) A Unified Framework for Numerically Inverting Laplace Transforms. *INFORMS Journal on Computing*, **18**, 408-421. http://dx.doi.org/10.1287/ijoc.1050.0137

[15] Abate, J. and Whitt, W. (1995) Numerical Inversion of Laplace Transforms of Probability Distributions. *ORSA Journal on Computing*, **7**, 36-43. http://dx.doi.org/10.1287/ijoc.1050.0137

[16] Craddock, M., Heath, D. and Platen, E. (2000) Numerical Inversion of Laplace Transforms: a Survey of Techniques with Applications to Derivative Pricing. *Journal of Computational Finance*, **4**, 57-82. http://dx.doi.org/10.21314/JCF.2000.055

Appendix

A.1. Euler Algorithm

```
function x = Euler(n, P, S, K, T, sigma)
%% Recommended parameters by Abate-Whitt (1995).
m = 11;
A = sym(18.4);

%% Parameters of the normalized option price.
h = sym((T*sigma^2)/4);
q = sym((K*T*sigma^2)/(4*S));
nu = sym(-2*log(P)/(sigma^2*T) - 1);

 %% Lambda calculation (Euler algorithm).
 arg = sym(zeros((n + m) + 1, 1));
 for k = 0:(n + m)
     arg(1 + k) = (A + 2*k*pi*1i)/(2*h);
 end

 if sum(single(abs(arg)) < max(0, 2*(single(nu) + 1))) ~= 0
    x = -999; % An easy to identify numeric error output.
    return
 end

 try
 %% Substitution into the Laplace transform.
 mu = sqrt(2*arg + nu^2);
 alpha = (mu + nu)/2;
 beta = (mu - nu)/2;

 integrand = @(alpha, beta, q, u) u.^(double(beta) - 2).* ...
     (1 - u).^(double(alpha) + 1).*exp(-u/(2*double(q)));
 a = sym(integral(@(u)integrand(alpha, beta, q, u), 0, 1, ...
     'ArrayValued', true, 'AbsTol', 0, 'RelTol', 1));
 a = log(a) + log(2*q)*(1-beta) - log(2.*arg) ...
     -log(alpha + 1) - mfun('lnGAMMA', double(beta));
 a = real(exp(a));

 %% Summation (Euler algorithm).
```

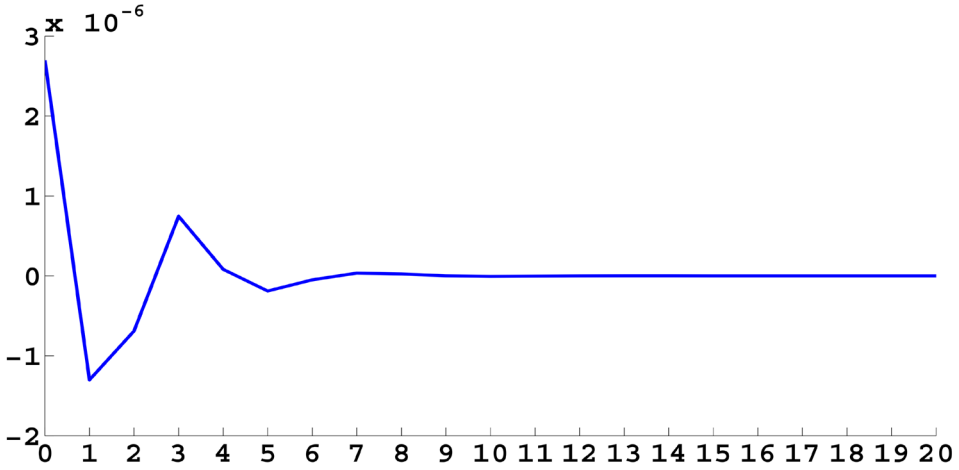
```
s = (exp(A/2)/h).*((−1).^(0:(n + m)))'.*a;
s(1) = s(1)/2;
for k = 1:(n+m)
    s(1 + k) = s(1 + k) + s(k);
end

x = sym(0);
for k = 0:m
    x = x + nchoosek(m, k)*2^(−m)*s(1 + (n + k));
end

x = x*P*(S*4)/(T*sigma^2);
x = double(x);

catch
    x = double(−999);
end

end
```

A.2. Talbot Algorithm

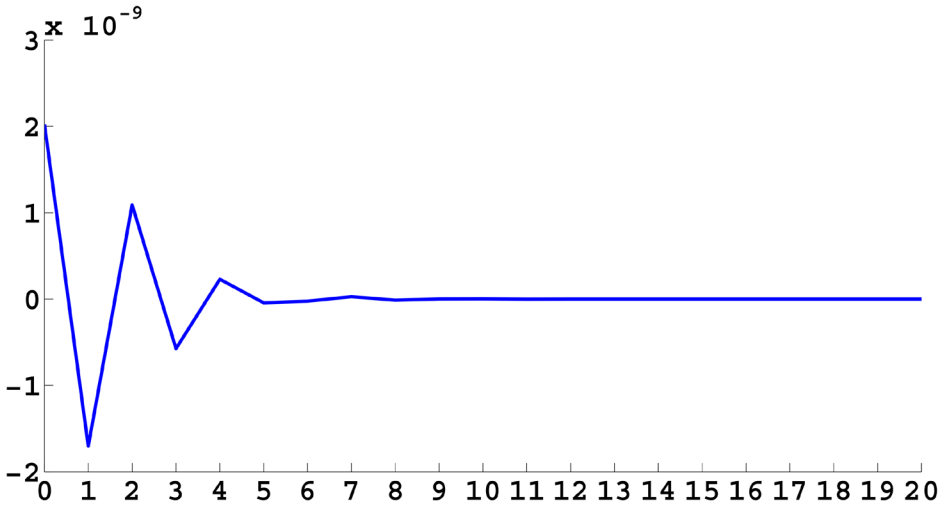
```
function x = Talbot(M, P, S, K, T, sigma)
%% Parameters of the normalized option price
h = sym((T*sigma^2)/4);
q = sym((K*T*sigma^2)/(4*S));
nu = sym(−2*log(P)/(sigma^2*T) − 1);

%% Lambda calculation (Talbot algorithm).
d = sym(zeros(M,1));
d(1) = 2*M/5;
for k = 1:(M − 1)
    d(k + 1) = 2*k*pi/5 * (cot(k*pi/M) + 1i);
end

g = sym(zeros(M,1));
g(1) = 1/2 * exp(d(1));
for k = 1:(M − 1)
    g(k + 1) = (1 + 1i*(k*pi/M)*(1 + (cot(k*pi/M))^2) ···
```

```
18              −1i*cot(k*pi/M))*exp(d(k+1));
19  end
20
21  arg = d/h;
22
23  if sum(single(abs(arg)) < max(0, 2*(single(nu) + 1))) ~= 0
24      x = −999; % An easy to identify numeric error output.
25      return
26  end
27
28  try
29  %% Substitution into the Laplace transform.
30  mu = sqrt(2*arg + nu^2);
31  alpha = (mu + nu)/2;
32  beta = (mu − nu)/2;
33
34  integrand = @(alpha, beta, q, u) u.^(double(beta) − 2).* ...
35        (1 − u).^(double(alpha) + 1).*exp(−u/(2*double(q)));
36  a = sym(integral(@(u)integrand(alpha,beta,q,u), 0, 1, ...
37      'ArrayValued', true, 'AbsTol', 0, 'RelTol', 100));
38  a = log(a) + log(2*q) * (1-beta) − log(2.*arg) ...
39      −log(alpha + 1) − mfun ('lnGAMMA', double(beta)) + log(g);
40  a = real(exp(a));
41
42  %% Summation (Talbot algorithm).
43  x = sym(0);
44  for k = 0:(M − 1)
45      x = x + a(k + 1);
46  end
47
48  x = x*2/(5*h)*P*(S*4)/(T*sigma^2);
49  x = double(x);
50
51  catch
52      x = double(−999);
53  end
54
55  end
```

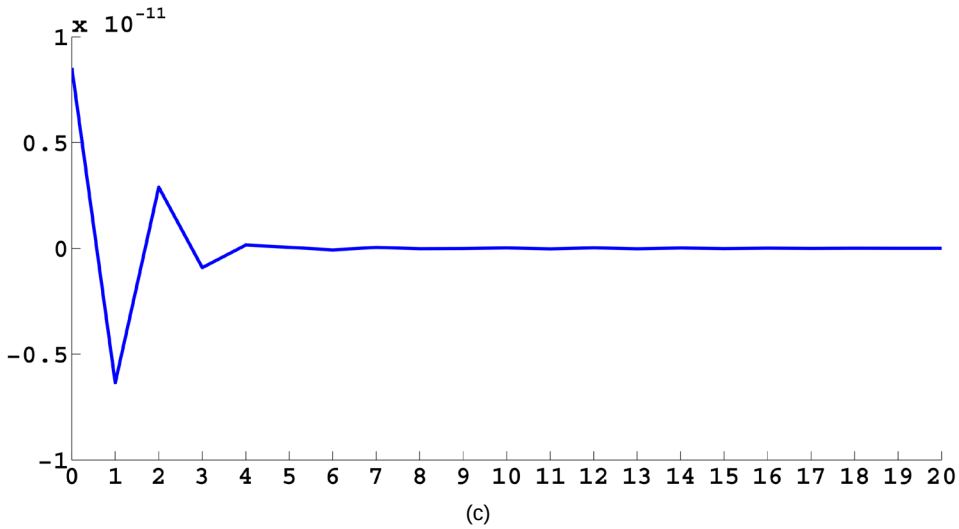
A.3. Behavior of the Laplace Transform



(a)

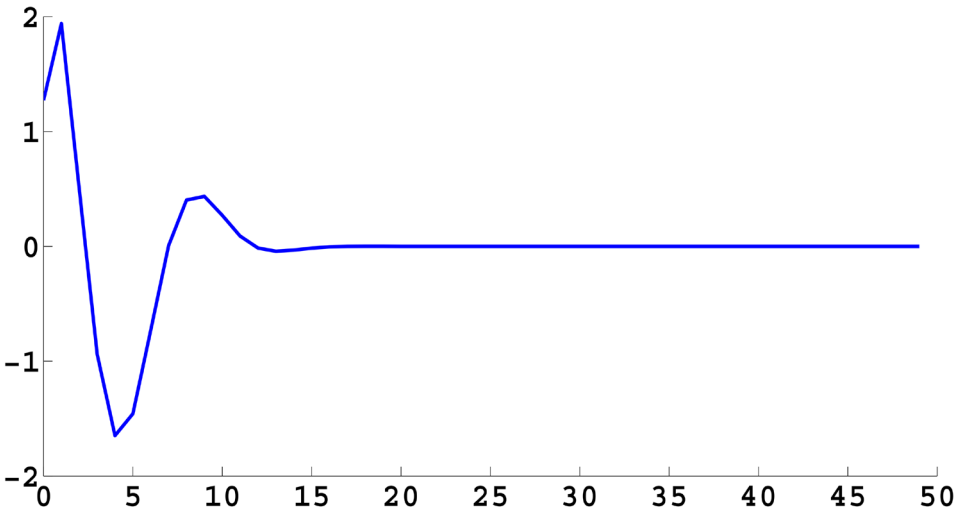


(b)

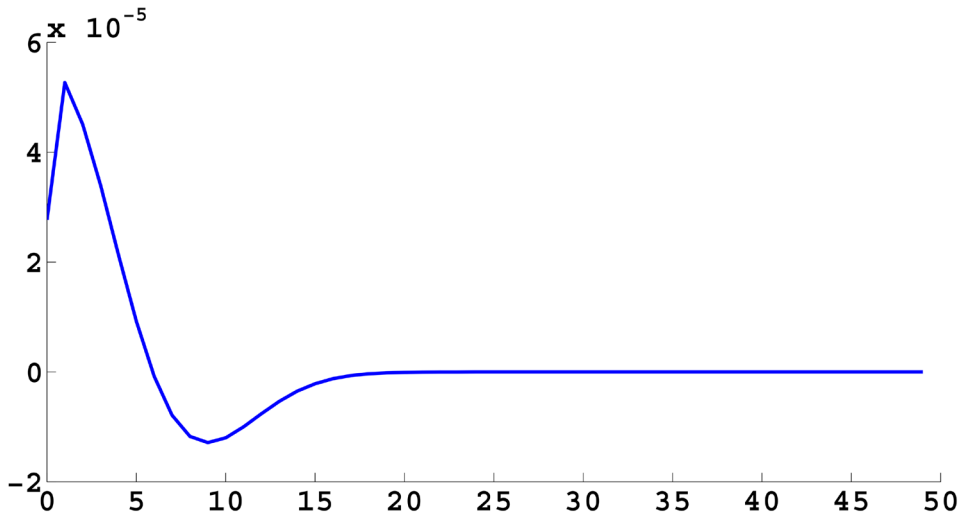


(c)

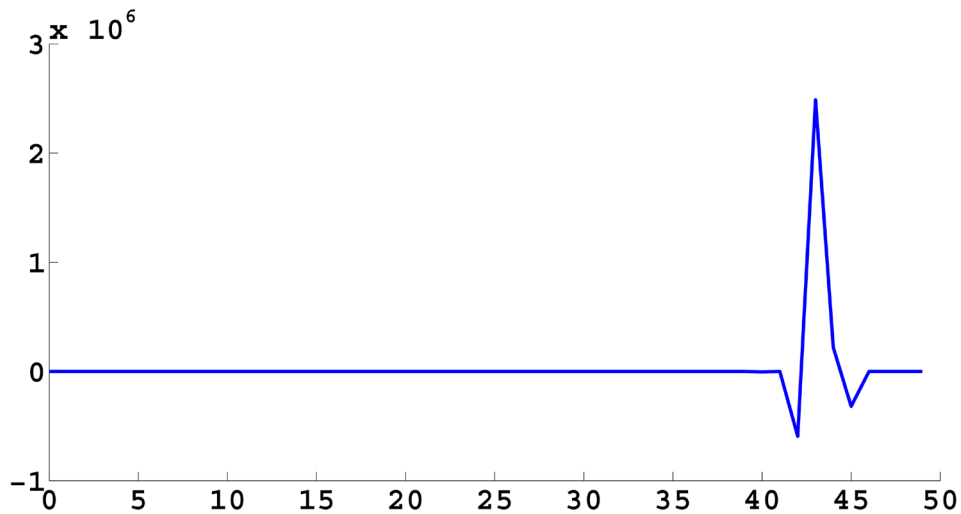
Figure A1. Magnitude of the Laplace transform at k using the Euler algorithm (n = 20).



(a)



(b)



(c)

Figure A2. Magnitude of the Laplace transform at k using the Talbot algorithm (M = 50).

A.4. Simulation Approach

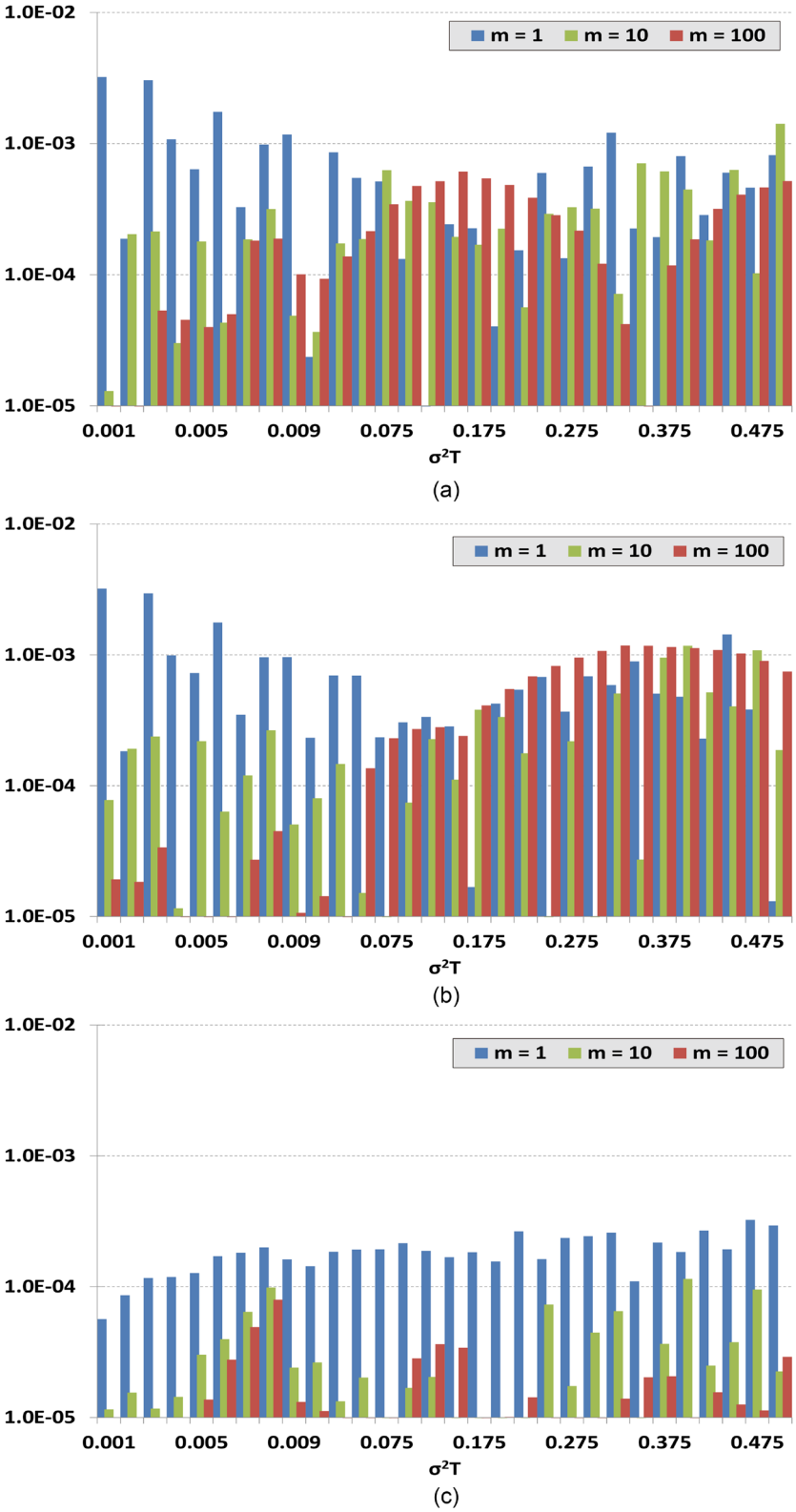


Figure A3. Absolute error of algorithms for different levels of m (n = 100 000).

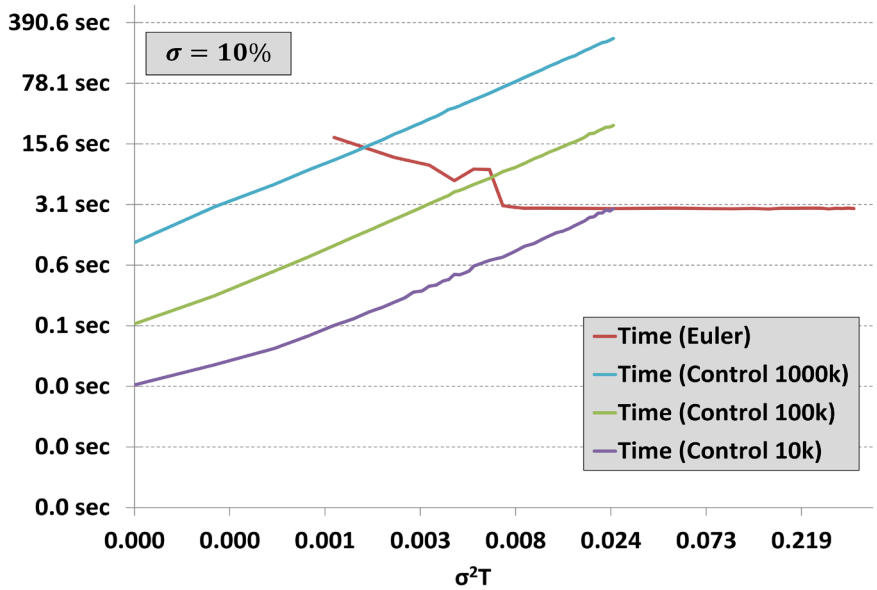


Figure A4. Comparison of calculation times using different methods (m = 10).

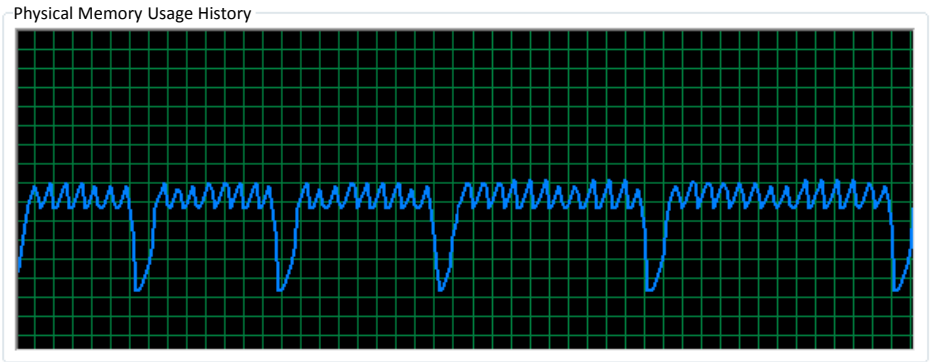


Figure A5. Optimized memory management of simulation algorithms.

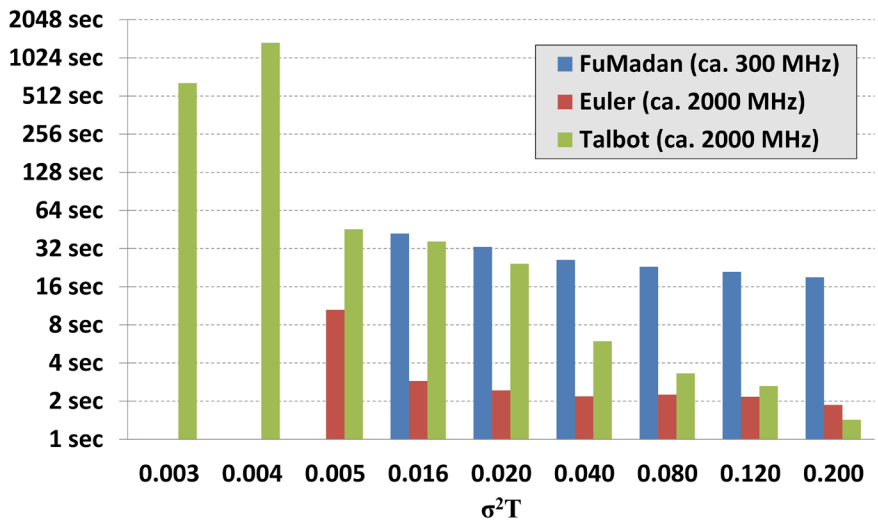


Figure A6. Speed improvement achieved using different methods.