Pricing of Discretely Monitored Barrier Options - Improvement of an Approximation Formula

Filip Andersson & Mikael Ögren

Master's thesis
2014:E29

Abstract

There are many different methods for pricing discretely monitored barrier options. There is a trade-off, however, between speed and accuracy. The players on the financial markets would of course ideally want a method which is both exact and returns a price instantaneously.

In this thesis we start from a fast, but on the other hand somewhat less accurate, approximation formula. It will be referred to as the *0.5826-approximation*, and was introduced in 1997 by Broadie, Glasserman and Kou [1]. It is one of the option pricing formulas currently used by SunGard. The idea of the 0.5826-approximation is to use the analytical pricing formula for the corresponding continuously monitored barrier option, and to use an adjusted barrier in that formula to account for the decreased probability of a barrier hit.

The purpose of this thesis is to improve the 0.5826-approximation for down-and-out call options with barrier less than or equal to the strike, and in particular to mitigate two problems. Firstly, the 0.5826-approximation performs less well when the underlying is close to the barrier (problem one). Secondly, an assumption of the formula is that the time to the first monitoring instant, Δt^{first}, is equal to the time steps between the other monitoring instants, Δt, (an assumption which is commonly violated in practice, problem two).

We propose two adjustments of the 0.5826-approximation that dramatically improve its overall performance (for down-and-out call options with barrier less than or equal to the strike) in the quite extensive tests that we have performed.

The first adjustment maintains the assumption that the time to the first monitoring instant is equal to the time steps between the other monitoring instants, and is based on the observation that the difference between the true and the 0.5826-approximated price is very similar to the time value of a vanilla option scaled by an appropriately chosen delta of the corresponding continuously monitored barrier option.

The second adjustment, which allows for $\Delta t^{first} < \Delta t$, is based on the fact that, in the Black-Scholes framework, the risk neutral probability of hitting the barrier at the first monitoring instant is known. The idea is to use this to weight two option prices, one corresponding to Δt^{first} and one corresponding to Δt. Both prices are (essentially) calculated using the first adjustment. We propose that this adjustment only be used when the value of the underlying is below the barrier.

Tests that have been performed show for instance that, when the proposed improvements of the 0.5826-approximation are used, the accuracy of the approximation is improved in 98 percent of the tested cases[1], both when the time between all monitoring instants is constant and when the time to the first monitoring instant is allowed to differ[2]. Since the tests focus on prices of the underlying close to the barrier, the proposed improvement of the 0.5826-approximation can be said to fill the purpose of this master's thesis.

[1] This percentages refers to the case when the cost of carry is equal to zero. This is the case with the least model dependency of the ones we examine. We obtain good results also for a cost of carry different from zero, but more model dependency is then introduced.

[2] In the case with a different time to the first monitoring instant, this percentage refers to strict improvements where the adjustment *is used*, that is below the barrier. Above the barrier the plain 0.5826-approximation is still used, implying of course that there will be equality in all cases.

Acknowledgements

We would like to thank our supervisor at SunGard, Samuel Sirén, for providing us with the subject of the master’s thesis, the problems that we seek to mitigate, and the initial hypotheses. We would like to thank Mr Sirén for his guidance throughout the work process.

We would like to thank our supervisor at LTH, Alexandros Sopasakis, for giving us guidance and support throughout the work process.

Table of Contents

1 Introduction

In this section we provide the reader with background information which will simplify the understanding of the rest of the thesis. We also establish the purpose of this thesis, provide a brief summary of our main conclusion, provide some notation, and outline the structure of the rest of the thesis.

1.1 Background information

1.1.1 Barrier options

A barrier option is an option whose payoff function changes *persistently* the first time the price of the underlying asset reaches a predetermined level – the ”barrier”.

There are barrier options of many different kinds. The type is always defined by a combination of the words in/out and up/down (for example “down-and-out call”) [2]. The in/out part determines what consequences a barrier hit will have – either that the option “comes into life” or “becomes worthless”. The option coming into life typically means that its payoff goes from being equal to zero regardless of the price of the underlying to being non-zero for certain values of the underlying. The option becoming worthless typically means that its value from then on is zero regardless of the price of the underlying. For some barrier options, however, the holder gets a *rebate* in the event of a barrier hit, meaning that the holder gets a cash amount paid out either immediately or at expiry[3]. The up/down part determines whether a price of the underlying which is above (up) or below the barrier (down) renders a barrier hit.

1.1.1.1 Difference between continuous and discrete monitoring

Barrier options can be either continuously or discretely monitored. From a theoretical point of view, the continuous case is more appealing since it is easier to derive pricing formulas in this case. In practice, however, discrete monitoring is also common.

In the continuous case, the barrier is considered hit if the price of the underlying reaches the barrier level at any point in time. In the discrete case, on the other hand, it is only checked at certain points in time (or in certain time intervals[4]) whether the barrier is hit. For a down-and-out barrier option for instance, it is checked whether $S \leq H$ at the monitoring instants, where S denotes the price of the underlying asset (for example a share) and H denotes the barrier. This is what we will refer to as a “barrier hit” for discrete barrier options throughout the thesis.

The probability of a barrier hit is of course lower for discretely monitored barrier options, since an “actual hit” might not “count” if it takes place when there is no monitoring.

1.1.1.2 Current methods for pricing of discretely monitored barrier options

In this section we describe the standard approximation formula, analytical and exact methods, and numerical methods for pricing discretely monitored barrier options. One may ask oneself why an approximation formula is needed in the first place. The reason is that calculating prices using the most accurate (sometimes exact) pricing methods in general requires relatively long computation times.

1.1.1.2.1 0.5826-approximation

In 1997, Broadie, Glasserman and Kou [1] presented an approximation formula, which we in this thesis often refer to as the *0.5826-approximation*, for the price of discretely monitored down-and-out/in calls and up-and-out/in puts where the barrier is shifted to reflect the lower probability of hitting the barrier compared to the continuous case (since the underlying can pass the barrier between

[3] Throughout the rest of this thesis we assume that there are no such rebates.
[4] Throughout the rest of this thesis we assume that monitoring is made not during intervals but rather at discrete “instants” or “points in time”.

monitoring points, compare section 1.1.1.1). This adjusted barrier, denoted H_A, is used in the continuous barrier option pricing formula to get an approximation of the price of the discretely monitored barrier option. The adjustment is done by shifting the barrier in such a way that the probability of a barrier hit decreases, specifically by multiplying the barrier by the factor $e^{\pm\beta\sigma\sqrt{\Delta t}}$, i.e. $H_A = H \cdot e^{\pm\beta\sigma\sqrt{\Delta t}}$ ("+" is used for up options and "−" for down options), where $\beta = -\zeta(\frac{1}{2})/\sqrt{2\pi} \approx 0.5826$ (with ζ the Riemann zeta function), and where σ denotes the volatility of the underlying. Δt, which denotes the time between monitoring instants, is assumed to be constant.

1.1.1.2.1.1 Extensions of the 0.5826-approximation

Kou has showed that the 0.5826-approximation can in fact be used also for up-and-out/in calls and down-and-out/in puts [3]. Hörfelt [4] has also expanded on the work by Broadie, Glasserman and Kou, and has presented (different) formulae for the cases (later) covered by Kou.

1.1.1.2.2 Analytical and exact methods

There exist analytical solutions for the price of discretely monitored barrier options [5]. To practically use these, however, it is in general necessary to evaluate multivariate normal distribution functions (the dimension of which is equal to the number of monitoring instants [1]) [5]. This is, ironically, done by numerical approximations (which become less accurate when the number of monitoring instants increase) [5]. Therefore, using these analytical solutions is not really an option for many monitoring instants.

Fusai, Abrahams, and Sgarra [6] presented an exact solution for the price of discretely monitored barrier options in 2006. The solution is based on the Black-Scholes framework where the authors have reduced the valuation problem to an analytically solvable scalar Wiener-Hopf integral equation. "The solution is obtained explicitly […] in terms of infinite sums of simple functions plus a single special function [6, p. 2]." This makes this approach substantially slower than the 0.5826-approximation method. When Fusai, Abrahams, and Sgarra ran their Wiener-Hopf solution code it took 50 seconds to get a three-digit accurate result. As a comparison, using the 0.5826-approximation only takes a few milliseconds.

1.1.1.2.3 Numerical methods

While not the focus of this thesis, it should be pointed out that there exist many different numerical methods for pricing discretely monitored barrier options.

Methods based on binomial and trinomial trees are common in the literature. Other approaches include numerical solutions of PDEs and numerical integration. An overview can be found in an article by Fusai and Recchioni [7].

1.1.2 The Black-Scholes framework[5]

In this thesis we assume that all options are of the European type and that a version of the "Black-Scholes framework" applies. We use a generalized Black-Scholes-Merton model with the cost of carry defined as $g = r - \delta$, where r denotes the risk free interest rate and δ denotes the continuous dividend yield, corresponding to Merton's extension of the Black-Scholes model [8, pp. 7-8]. In this version of the framework, the Q-dynamics (i.e. the dynamics under the risk neutral measure) of S are given by [9, p. 238]

$$dS_t = gS_t dt + \sigma S_t dW_t. \tag{1.1}$$

[5] We assume that the reader is familiar with the formulae for pricing European style options in this setting. A possible reference is otherwise [8].

S_t denotes the price of the underlying asset at time t[6], σ the volatility of the underlying asset, and W_t a Q-Wiener process [9, p. 103].

1.2 Purpose

The purpose of this thesis is to improve the 0.5826-approximation formula for down-and-out call options with barrier less than or equal to the strike, and in particular to improve its performance when the underlying is close to the (adjusted) barrier (see further the specific problems introduced in section *2 Problem definition and delimitation*).

1.3 Conclusion in brief

We suggest that the following adjustment of the 0.5826-approximation formula, which provides the approximated price, V, of a discretely monitored down-and-out call option[7], be used[8]:

$$V = 1_{S/H<1} \cdot \left(kV^{eq} + (1-k)V^{first}\right) + \left(1 - 1_{S/H<1}\right) \cdot C_{DO}^{0.5826}(T,S,X,H,\sigma,r,g,\Delta t), \tag{1.2}$$

where V^{eq} is the approximated price (using the method introduced in section 4 of this thesis) of a discretely monitored down-and-out call option with time between monitoring instants Δt (and with $\Delta t^{first} = \Delta t$, where Δt^{first} denotes the time to the first monitoring instant). V^{first} is the approximated price using the same method, but with the time between *all* monitoring instants equal to Δt^{first} (although the delta is the same as for V^{eq}). $1_{S/H<1}$ denotes an indicator function. Precise formulation for those variables as well as k is provided below,

$$k = C_{digital}\left(\hat{T}_{digital}, S, \hat{X}_{digital}, \sigma, \hat{r}, g\right),$$

$$V^{eq} = C_{DO}^{0.5826}(T,S,X,H,\sigma,r,g,\Delta t) + \Delta(H_A) \cdot \left(C\left(\hat{T}, S, \hat{X}, \sigma, \hat{r}, g\right) - max\left(S - \hat{X}, 0\right)\right),$$

$$V^{first} = C_{DO}^{0.5826}\left(T,S,X,H,\sigma,r,g,\Delta t^{first}\right) + \Delta(H_A) \cdot \left(C\left(\hat{T}^{first}, S, \hat{X}^{first}, \sigma, \hat{r}, g\right) - max\left(S - \hat{X}^{first}, 0\right)\right),$$

where

$$1_{S/H<1} = \begin{cases} 1, & if \quad S/H < 1 \\ 0, & if \quad S/H \geq 1, \end{cases}$$

$$\hat{T}_{digital} = \gamma \Delta t^{first},$$
$$\hat{X}_{digital} = H,$$
$$\hat{r} = 0,$$
$$\hat{T} = \alpha \Delta t,$$
$$H_A = He^{-0.5826\sigma\sqrt{\Delta t}},$$
$$H_A^{first} = He^{-0.5826\sigma\sqrt{\Delta t^{first}}}$$
$$\hat{X} = H_A,$$
$$\hat{T}^{first} = \alpha \Delta t^{first},$$
$$\hat{X}^{first} = H_A^{first},$$

[6] Since we only consider prices of the underlying at time zero, i.e. S_0, in this thesis, for simplicity we define $S = S_t, t = 0$. This notation is used throughout the thesis.

[7] The formula has in this thesis exclusively been tested on down-and-out call options with $H \leq X$.

[8] It should be pointed out that we find a separate formula in the case of $\Delta t^{first} = \Delta t$, presented for instance in section 4.4.

$$\Delta(H_A) = \frac{\partial C_{DO}^{0.5826}}{\partial S}(H_A).$$

We recommend using[9]

$$\alpha = 0.733,$$
$$\gamma = 1.41.$$

H_A denotes the adjusted barrier used for pricing a discretely monitored down-and-out call with the 0.5826-approximation, using Δt as the time between monitoring instants. H_A^{first} is identical to H_A except for the fact that Δt^{first} is used instead of Δt.

$C_{digital}$ denotes the price of a digital call option[10] and $C_{DO}^{0.5826}$ denotes the 0.5826-approximated price of a discretely monitored down-and-out call option. $\Delta(H_A)$ is calculated as the partial derivative given above[11], and is the delta of the 0.5826-approximated price of the discretely monitored down-and-out call option at the adjusted barrier, H_A.

There exist fairly standardized notation in mathematical finance, which we are certain the reader is familiar with. We have tried to use this to as large an extent as possible. Various accents, superscripts etcetera are used to show differences between different parameters, but in general the "basic letter" always has the same fundamental meaning. Since this is the first time in the thesis that a lot of formulae and notation is introduced, in the next paragraph we provide the meaning of the most common notation ("basic letters") as a service to the reader.

T denotes the time to maturity of an option, S the price of an underlying asset, X the strike (or exercise) price of an option[12], H the barrier level of a barrier option, σ the volatility of an underlying asset, r the risk free interest rate, g the cost of carry, and Δt the time between monitoring instants for barrier options.

These improvements are achieved when using (1.2)[13]:

1. Improved accuracy close to the (adjusted) barrier.
2. Improved accuracy when the time to the first monitoring instant is less than the time between the other monitoring instants ($\Delta t^{first} < \Delta t$).

1.4 Notation

Notation will in general be introduced when it is first used, and an overview of all notation can be found in *Appendix A: Notation*.

All parameters and variables in bold are in matrix or vector form. If nothing else is stated, these have a number of rows equal to the number of simulations, n.

An asterisk, $*$, that is not in superscript, denotes an element-by-element multiplication[14] of two vectors or matrices (with the same dimensions), that produces a new vector or matrix with those same dimensions.

[9] These coefficient values have been found to be optimal in the case of zero cost of carry ($g = 0$). We have also found values in the case when g is different from zero. These will be presented in the thesis, but we do not consider them part of our main result due to additional model dependency introduced in this case.
[10] It should be noted that the digital option used is a cash-or-nothing call that pays out either one currency unit or nothing at maturity (depending of course on whether or not it is in the money).
[11] The partial derivative of an option price with respect to the price of the underlying asset is generally referred to as the "option delta". So also in this thesis.
[12] Note specifically that we throughout the thesis use X rather than K, which is also common in the literature, to denote the strike price.
[13] Compare to section *2 Problem definition and delimitation.*

The optimal α and γ have subscript g and r that are either zero or non-zero depending on whether or not they are calculated from data with simulated g and r. For example $\alpha_{g=0,r=0}^{n}$ is calculated from data with $g = 0$ and $r = 0$. If it is a γ it is calculated with the corresponding α. For example $\gamma_{\mathrm{g}\neq0,\mathrm{r}\neq0}^{n}$ is calculated with $\alpha_{g\neq0,r\neq0}^{n}$ from data with simulated g and r. When there is a superscript $b = 1$ the limit of S/H above which our adjustment is not used is equal to one.

1.5 Structure of the rest of the thesis

In section 2 we introduce the specific problems that will be addressed in this thesis.

Sections 4 and 5 are completely devoted to the first and second problems, respectively. Both these sections have the following overall structure:

1. Problem description
2. Hypothesis
3. Analysis and results
4. Conclusion

The thesis ends with a summarizing and concluding section.

[14] An obvious exception is the Matlab code in *Appendix B: Data simulation*.

2 Problem definition and delimitation

We examine (and try to mitigate) these problems for down-and-out call options[15]:

1. Approximation inaccurate when underlying close to (adjusted) barrier
2. Approximation inaccurate when $\Delta t^{first} \neq \Delta t$[16]

There may of course be other specific problems, but we focus *exclusively* on the ones listed above. The problems will be described in the rest of this section and also in sections 4.1 and 5.1, respectively.

The first problem is mentioned in the article where the 0.5826-approximation is presented [1]. The second problem is very natural since an assumption of the 0.5826-approximation formula is that the time between monitoring instants is constant.

In Figure 1 these problems are illustrated. We see that the 0.5826-approximation is not very accurate close to the (adjusted) barrier, although it is more accurate than simply using the unadjusted corresponding continuously monitored barrier option. We can also note that the price curve for the option with $\Delta t^{first} \neq \Delta t$ differs from the approximation, although in this specific case in general not as much as the one with $\Delta t^{first} \neq \Delta t$. In Figure 2 we see that for values of the underlying far away from the barrier the 0.5826-approximation is very accurate in both cases.

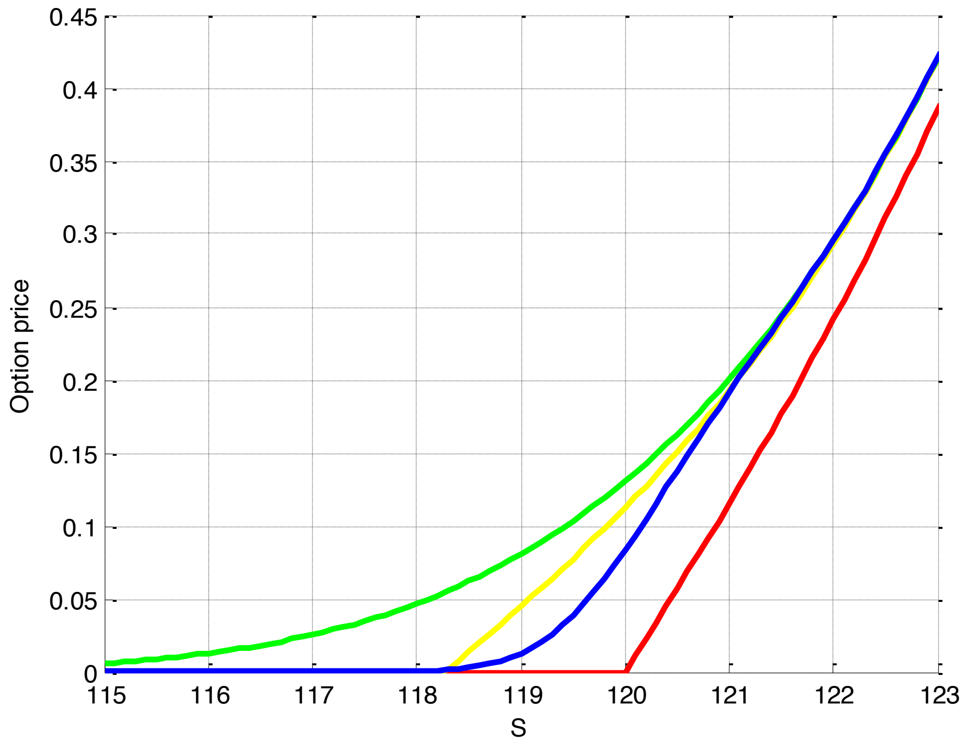

Figure 1: The green curve shows the true price of a discretely monitored down-and-out call option (with $\Delta t^{first} = \Delta t$), the blue curve the true price of the (approximately corresponding) option with $\Delta t^{first} \neq \Delta t$, the yellow curve the 0.5826-approximated price, and the red curve the price of the corresponding continuously monitored option, for different prices of the underlying asset, S. The following parameter values have been used: $X = 130, g = r = 0, \sigma = 0.1, T = 0.25, H = 120, \Delta t \approx 0.0615, \Delta t^{first} = 0.25/65$. This corresponds to $H_A \approx 118.3$.[17]

[15] We will furthermore exclusively consider options with $H \leq X$.
[16] We will only consider $\Delta t^{first} < \Delta t$.
[17] The attentive reader may notice that the number of monitoring instants will necessarily differ in the two cases. This is an inconsistency we have to accept given the nature of the approximation formula.

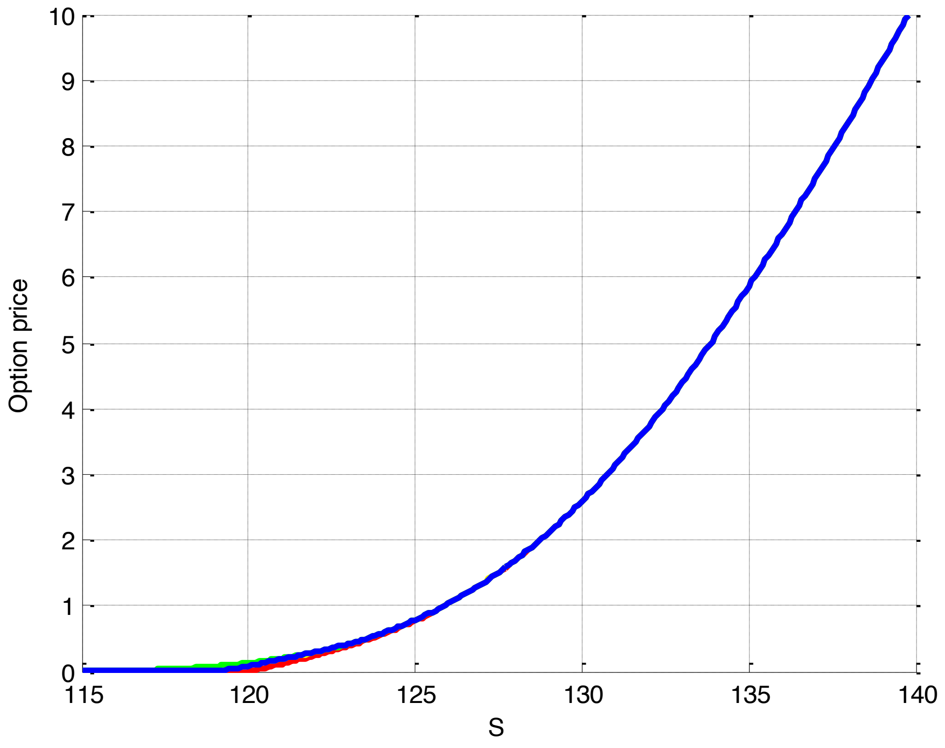

Figure 2: This is a zoomed out version of Figure 1. Color meanings and parameter values are thus identical.

Figure 3 below indicates that the 0.5826-approximation is in fact in general less accurate close to the adjusted barrier than elsewhere.

Figure 4 indicates that there is a problem also when $\Delta t^{first} < \Delta t$. This seems to be more prevalent close to the *unadjusted* barrier, and has a different character in that the error changes sign.

The difference between Figure 3 and Figure 4 makes it clear that a separate treatment of the case when $\Delta t^{first} \neq \Delta t$ is needed[18]. It can also be pointed out that this case is in fact very common (see section 5.1).

[18] We exclusively consider $\Delta t^{first} < \Delta t$ in this thesis.

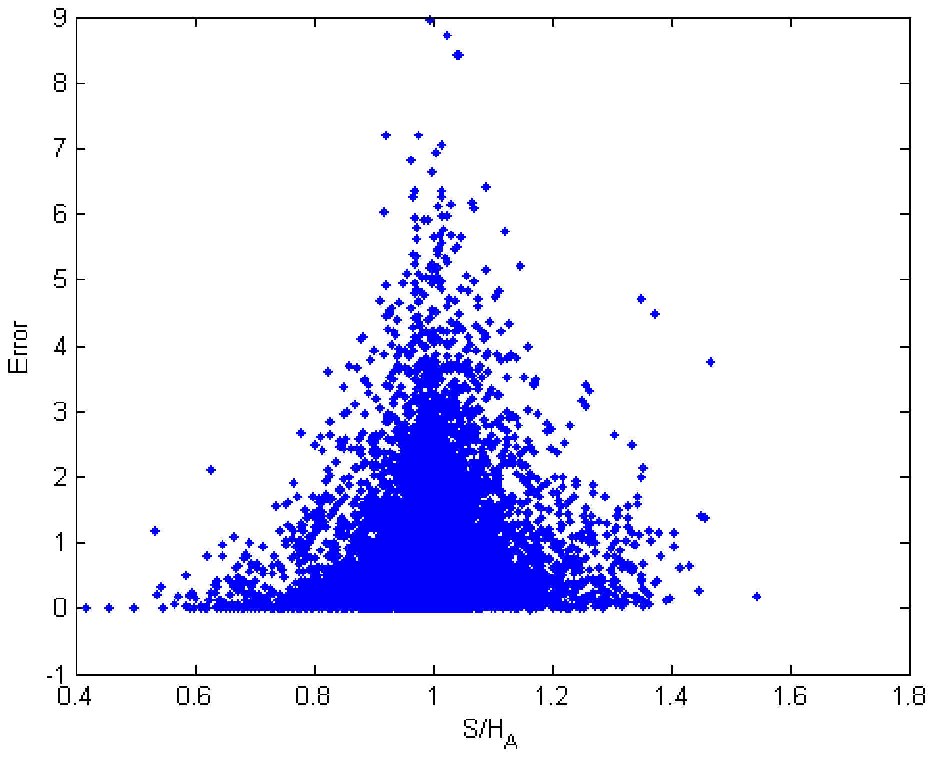

Figure 3: Difference between the true value and the 0.5826-approximated price of discretely monitored down-and-out call options plotted against $\frac{S}{H_A}$. *1,000 simulated parameter sets with normally distributed* S. $r = g = 0$, $\Delta t^{first} = \Delta t$.

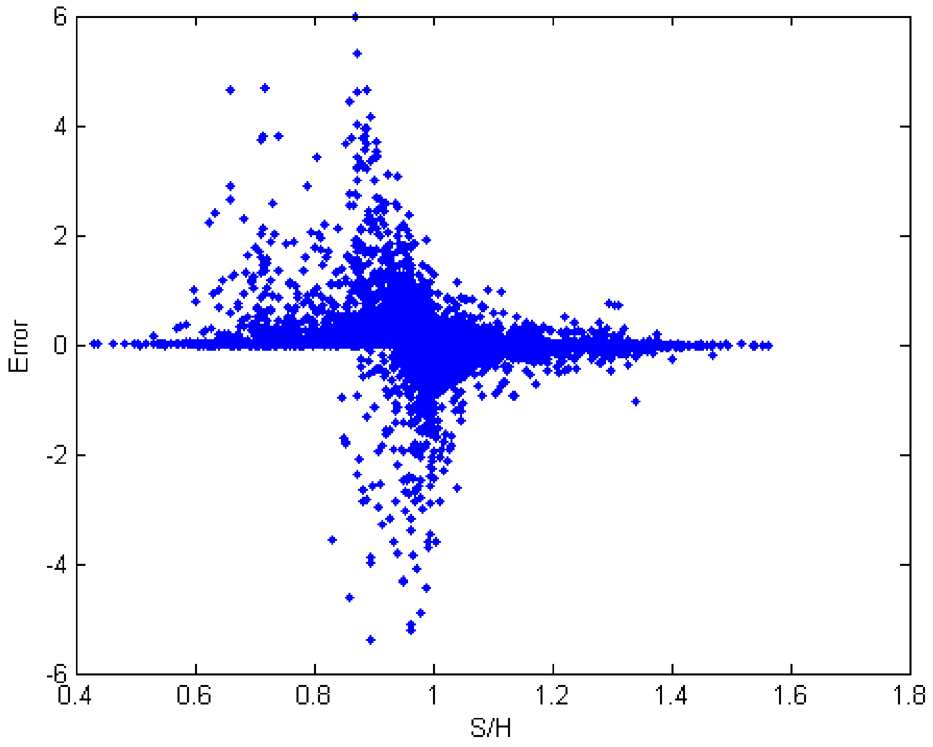

Figure 4: Difference between the true value and the 0.5826-approximated price of discretely monitored down-and-out call options plotted against $\frac{S}{H}$. *1,000 simulated parameter sets with normally distributed* S. $r = 0, g = 0, \Delta t^{first} < \Delta t$.

3 Data simulation

Our main tool for mitigating the problems outlined in the previous section is numerical optimization. It is thus necessary to have a large amount of data. This data has been simulated. The data simulation is obviously fundamental for proceeding with the analyses necessary to come to a conclusion, and has been quite a time consuming process. At the same time, it is not in itself a very *important* contribution of this thesis. We have therefore decided to move it to *Appendix B: Data simulation*, and we kindly refer the interested reader to that section. It should be pointed out that we assume in the following that the reader has read that appendix.

4 Problem 1: Approximation inaccurate close to barrier

The aim of this section is to improve the 0.5826-approximation in cases with $\Delta t^{first} = \Delta t$[19], in particular close to the (adjusted) barrier.

4.1 Problem description

As shown in Figure 5, the plain 0.5826-approximation produces an error compared to the true price of the discretely monitored barrier option, especially close to the (adjusted) barrier.

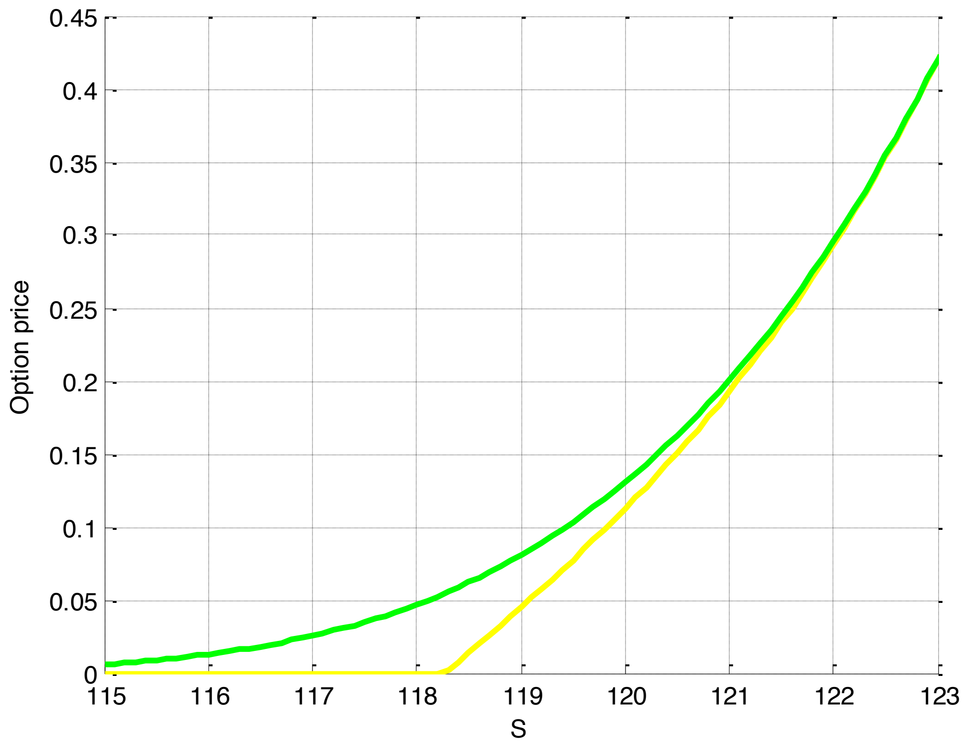

Figure 5: The green curve shows the true price of a discretely monitored down-and-out call option (with $\Delta t^{first} = \Delta t$) and the yellow curve depicts the 0.5826-approximated price for different prices of the underlying asset, S. The following parameter values have been used: $X = 130, g = r = 0, \sigma = 0.1, T = 0.25, H = 120, \Delta t \approx 0.0615$. This corresponds to $H_A \approx 118.3$.

4.2 Hypothesis

The hypothesized solution is to add an adjustment equal to the time value of an appropriately chosen vanilla call option scaled by the delta of a closely related barrier option to the approximated price obtained with the plain 0.5826-approximation. The full hypothesis is

$$V^{eq} = C_{DO}^{0.5826}(T, S, X, H, \sigma, r, g, \Delta t) + adjustment, \tag{4.1}$$

where

$$adjustment = \Delta(H_A) \cdot \left(C(\hat{T}, S, \hat{X}, \sigma, \hat{r}, g) - max(S - \hat{X}, 0)\right), \tag{4.2}$$

with

$$\hat{r} = 0,$$
$$\hat{T} = \alpha \Delta t,$$

[19] We may thus without loss of generality set $k = 1$ in the formula presented in section 1.3.

$$\hat{X} = H_A = He^{-0.5826\sigma\sqrt{\Delta t}},$$
$$\Delta(H_A) = \frac{\partial}{\partial S} C_{DO}^{0.5826}(H_A).$$

The hypothesis may be easier understood by studying Figure 6 and Figure 7. β_1 denotes the difference (error) between the true value of the discretely monitored barrier option and the 0.5826-approximation. β_2 denotes the time value of the vanilla option and the proposed adjustment, once again, is to multiply β_2 by the slope of the orange line in Figure 7, $\Delta(H_A)$. This calculation can be done for any value of the underlying, for example at the blue lines in the figures instead. It can be noted that the risk free interest rate for the vanilla call is equal to zero according to the hypothesis, regardless of the value of the true risk free interest rate.

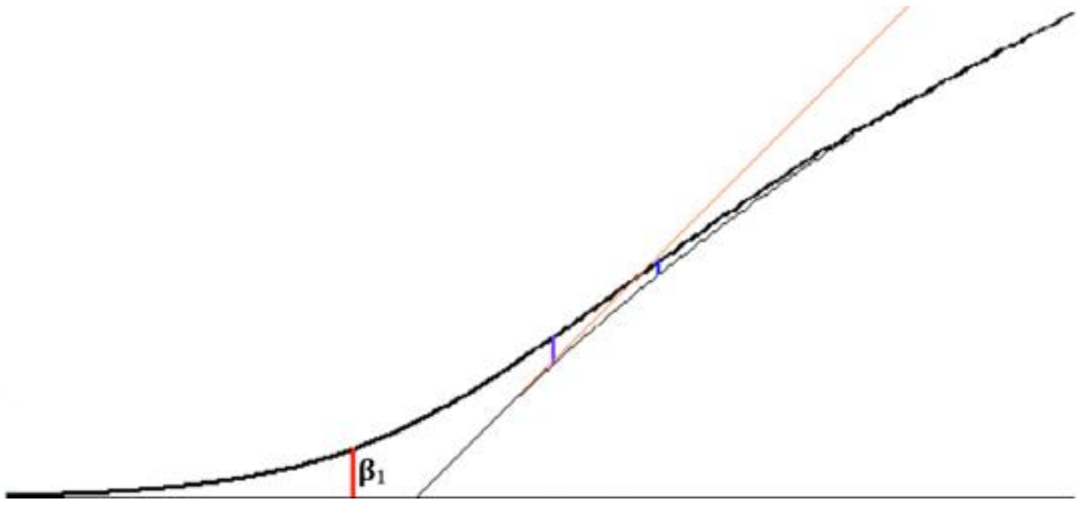

Figure 6: This is Figure 5 with a few additional elements. The essential addition is the orange line. Its slope coefficient equals $\Delta(H_A)$. The red line illustrates the difference between the true option value and the 0.5826-approximation for an arbitrary value of the underlying asset. It will be referred to as β_1. The blue details are only included for illustrative purposes (which will be clear in the text that follows).

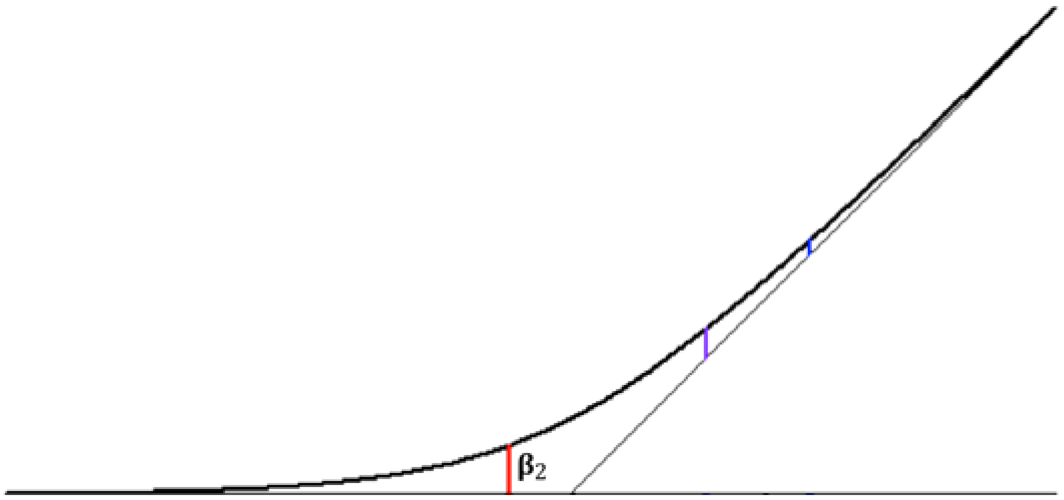

Figure 7: The thick black curve shows the price of a vanilla call option with strike H_A, time to maturity $\hat{T}$, and $r = 0$. Its intrinsic value is also plotted. The red line illustrates the time value of this option for an arbitrary value of the underlying asset. It will be referred to as β_2. The blue details are only included for illustrative purposes (which will be clear in the text that follows).

4.2.1 Rationale for the hypothesis

The idea to use the ansatz $\hat{T} = \alpha \Delta t$ is based on the fact that there is a relationship between the optimal time to maturity[20] of the vanilla call option in (4.2) and Δt (see further the test in section *4.3.1.1 Suitability of ansatz* that shows just this).

To use the call option in some way for minimizing the approximation error is very intuitive thanks to the great similarity of Figure 6 and Figure 7.

The delta of the 0.5826-approximated price of the discretely monitored down-and-out call option (calculated at the adjusted barrier), $\Delta(H_A)$, is used as a scale factor in the adjustment described above. To explain why it is appropriate to include it, we will in fact need to assume that this observation is correct and that, when the delta of the continuously monitored barrier option at H_A is equal to one, the errors between the true option values and the 0.5826-approximated values are exactly equal to the time values of the vanilla call (chosen as in (4.2)) for all values of the underlying asset.

When the delta of the barrier option is different from one, all else equal, we can then "reconstruct" the error by multiplying the time value of the vanilla call by the delta (see also Figure 8). The reason is that, since nothing is changed compared with the "delta one" case, we can regard the discretely monitored barrier option as a portfolio consisting of a (real) number of the instrument with delta equal to one. Specifically this means that the error is the same (real) number multiplied by the error from the case with delta equal to one.

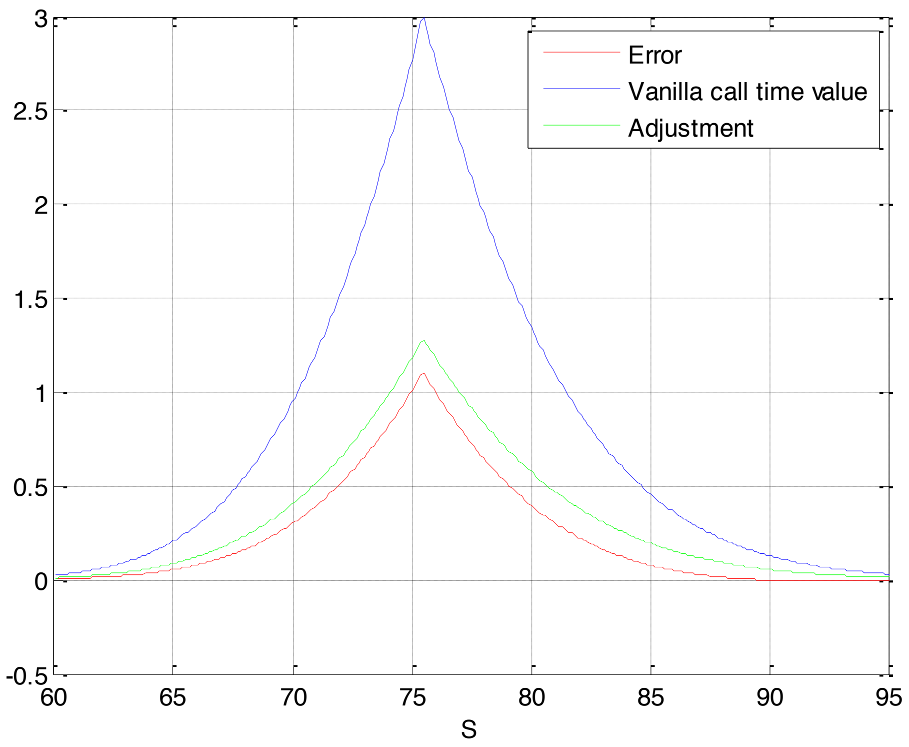

Figure 8: The red curve shows the error between the benchmark ("true") value and the 0.5826-approximation. The blue curve depicts the time value of the appropriately chosen vanilla call option. The green curve shows the hypothesized adjustment (4.2), which would in the ideal case be exactly equal to the error. Parameter values: $X = 100, g = 0, r = 0, \sigma = 0.1, T = 10, n_{monit}^{eq} = 10, H = 80$. This gives $H_A \approx 75.5$. In this example case we use $\alpha = 1$, equivalent to a time to maturity of Δt for the vanilla call.

[20] Optimal in the sense that it optimizes the accuracy of the approximation.

It is seen in Figure 8 that the approximation performs fairly well at least in one example case, with an as of yet unoptimized value of α.[21]

It was stated above that the vanilla call option should have a risk free interest rate equal to zero. Our reason for this is that we want to avoid "double discounting" of its price. The fact is that both the price of the vanilla call and the delta of the continuously monitored barrier option contain discounting factors equal to e^{-rT}. The 0.5826-approximation contains this factor as well, and the true price should change by this factor when we let r be different from zero. We want to maintain this property, which is of course done by avoiding the double discounting (i.e. by avoiding *another* multiplication of one (not all) of the terms by the factor e^{-rT}. It will be verified section *4.3 Analysis and results* that the optimal α does in fact seem to be independent of r when we avoid double discounting by setting the risk free rate for the vanilla call to zero.

4.3 Analysis and results

In this section we first make some initial tests that we consider necessary before performing our main analyses. We then move on to the main analyses needed to find an optimal value of α, test the obtained optimal values in different ways, and end the section by testing potential further improvements of our refined approximation formula.

4.3.1 Initial tests

We begin this section by performing two tests to examine whether the hypothesis seems reasonable. This is mostly done to assure ourselves that we do not proceed completely in vain with the rest of the analysis. The accuracy of the proposed adjustment with our final, optimal value of α will of course be the ultimate test. We first verify that the ansatz for the time to maturity of the vanilla call option makes sense, and then validate that the optimal α is in fact independent of r. We go on to check whether the optimal value of α is independent of g too, and finally seek indications that the benchmark pricing function that we use is accurate.

4.3.1.1 Suitability of ansatz for $\hat{T}$

To verify that the ansatz $\hat{T} = \alpha\Delta t$ makes sense, we examine the dependence between the optimal time to maturity for the call option in the adjustment term (in the rest of the current section denoted T^*[22]) and different parameters.

There seems to be no clear dependence between T^* and most of the parameters. As an example we show T^* plotted against H in Figure 9.

[21] We come back to this example in section 4.3.3.1.1, then with an optimized value of α.

[22] T^* is determined by solving the minimization problem described in section 4.3.2.2.2 *Minimization of absolute error for each simulated parameter set*, but with the optimization done with respect to T for the call option instead of with respect to α, i.e. no ansatz involving Δt has been made.

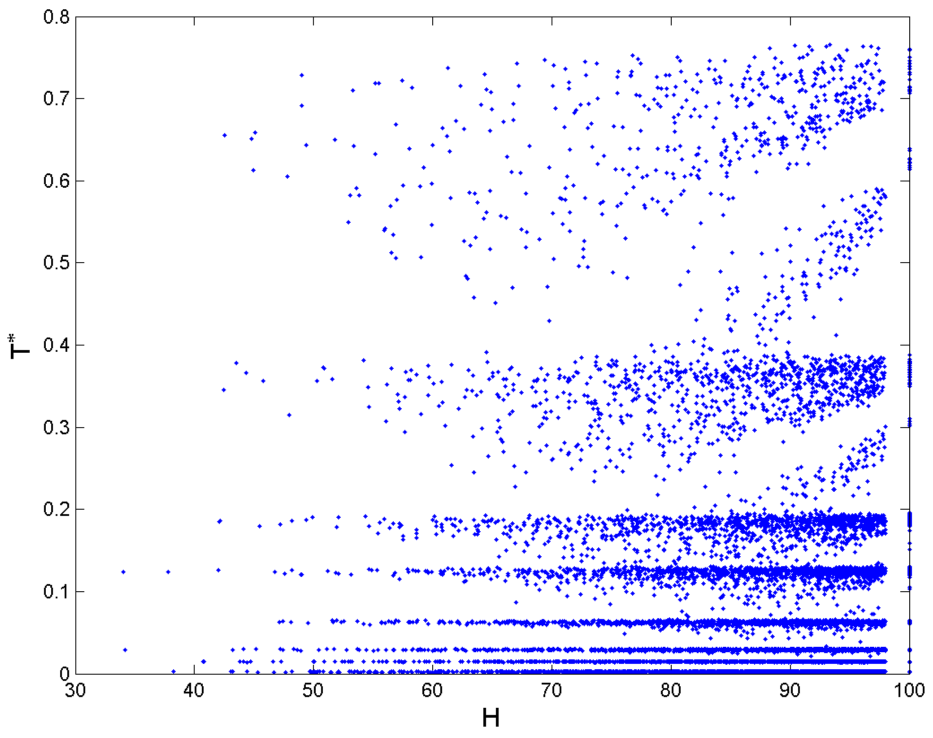

Figure 9: T plotted against H. Normally distributed S, 10,000 simulated parameter sets.*

The hypothesis that T^* depends on Δt, however, seems to be correct as can be seen in Figure 10.

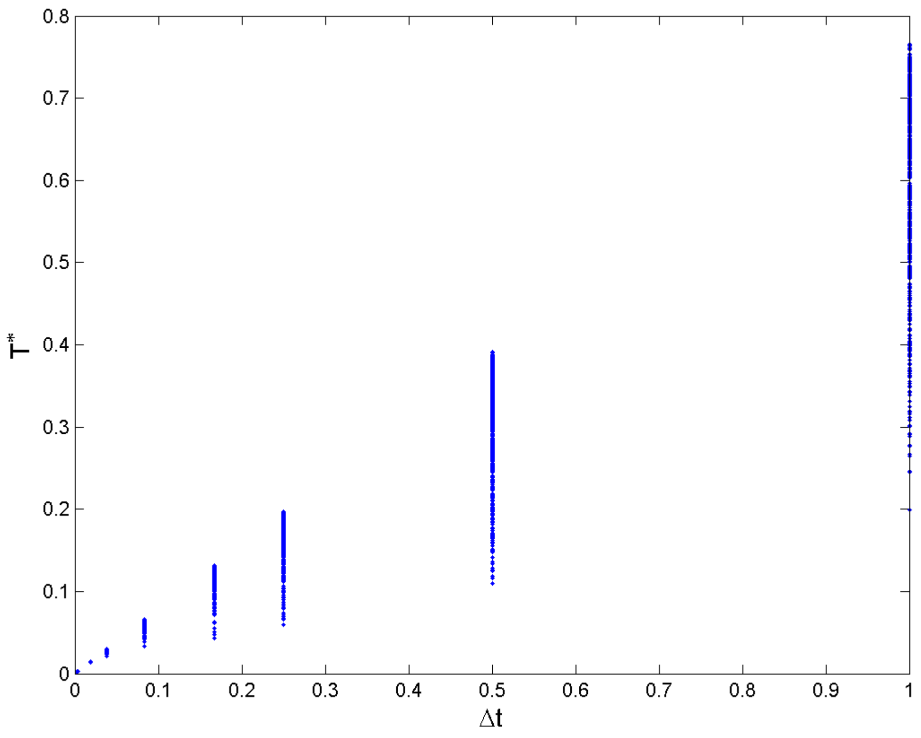

Figure 10: T plotted against Δt. Normally distributed S, 10,000 simulated parameter sets.*

4.3.1.2 Independence of r

Letting the risk free rate be equal to zero for the vanilla call option should (according to the previously made argument about “double discounting”, compare section 4.2.1) imply that, when the true r changes, the optimal value of α does not change at all. This is because a change in the true r will only give rise to a multiplication by $e^{-\Delta rT}$ of the true option value and our adjusted 0.5826-approximation alike, where Δr denotes the change in r. It should be verified that the relations between the benchmark value, the delta, and the 0.5826-approximation remain the same when the value of r is changed. Since the two option values can be zero, we can only have these in the numerator when performing such a test. This gives the following two relations to examine:

- $C_{DO}^{FD}/\Delta(H_A)$
- $C_{DO}^{0.5826}/\Delta(H_A)$

Testing shows that the maximum absolute difference between the cases when $r = 0.05$ and $r = 0$ is in the order of 10^{-13} for the two relations respectively. The corresponding differences between the cases when $r = 0.2$ and $r = 0$ are in the order of 10^{-12}. We assume these differences to be results of machine precision and conclude that the value of α is independent of r, and that it thus does not matter if we have r equal to, or different from, zero in our tests.

4.3.1.3 Dependence of g

It was shown in section 4.3.1.2 that r does not affect α. Since the cost of carry is arguably a close relative to the risk free rate of interest, a relevant question may be whether α is independent of the cost of carry as well. This is, at least to us, less self-evident. We will therefore analyze this in the following.

In Figure 11 the absolute values of the adjustment errors (i.e. the absolute values of the differences between the true values and the values obtained from the adjusted 0.5826-approximation) are plotted. The optimal α obtained for $g = 0$ is used when calculating the adjustment errors for other values of g. Thus, if α were independent of g, no deviations in adjustment errors would be seen between the series since the simulated parameter values are the same for all series. It is clearly seen in Figure 11 and Figure 12 that this is not the case. On the contrary, the adjustment errors generally get larger for larger values of g. The cost of carry can thus *not* be disregarded in further analyses.

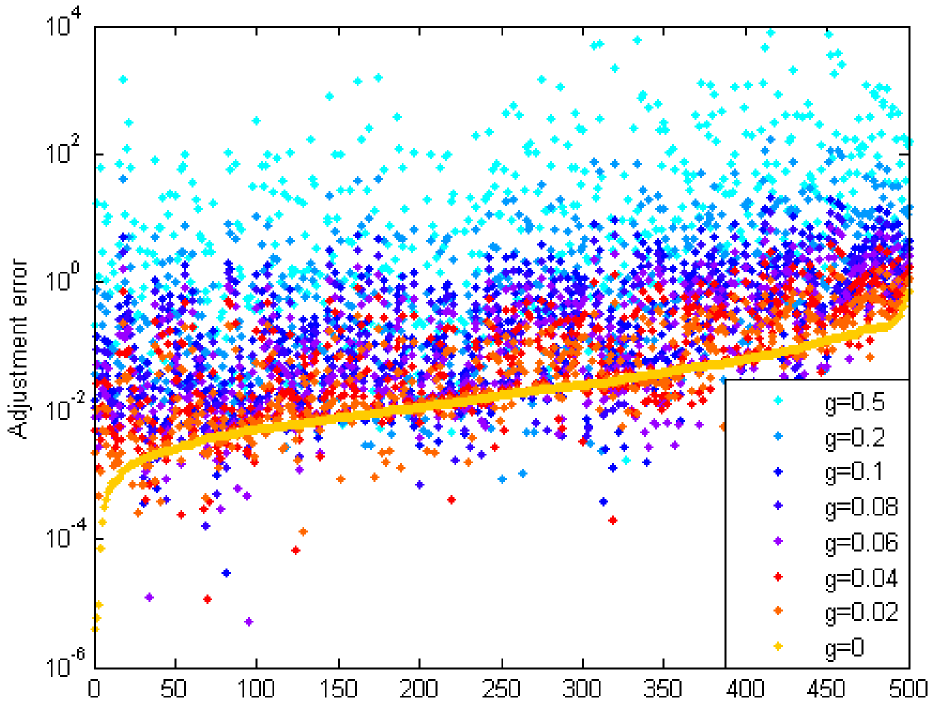

Figure 11: One of the eleven simulated S-vectors (normally distributed). The absolute adjustment errors (true value minus our adjusted 0.5826-approximation) in the case of $g = 0$ (and with the corresponding optimal $\alpha, \alpha^n_{g=0,r=0}$) are sorted in ascending order for 500 simulated parameter sets, and the corresponding error is then plotted for the other values of g for each of these parameter sets (using $\alpha^n_{g=0,r=0}$).

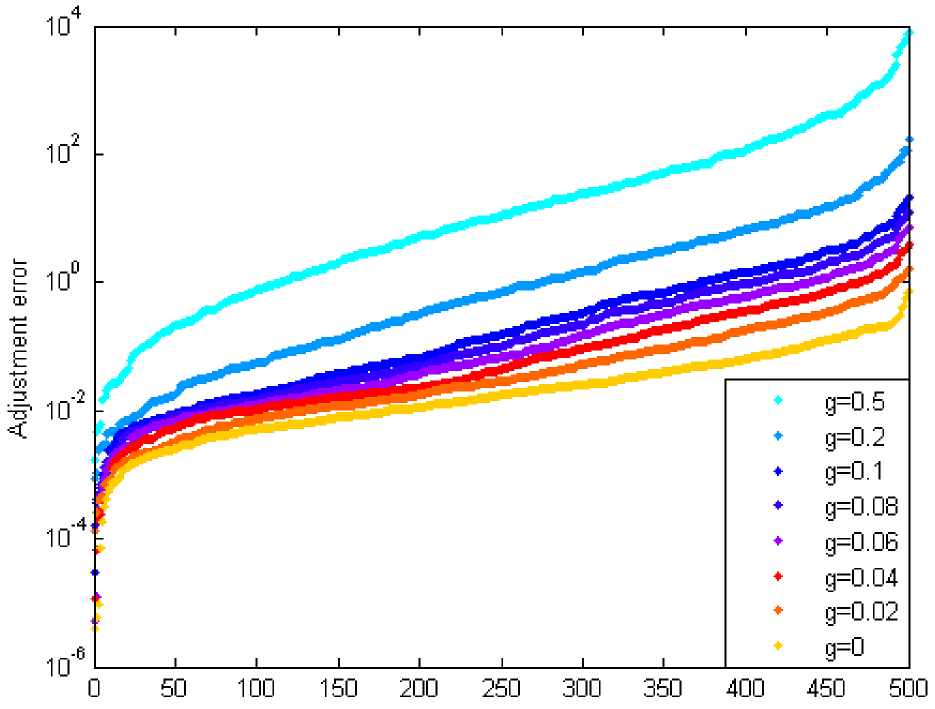

Figure 12: Absolute adjustment errors sorted in ascending order, for different values of g, for one of the eleven normally distributed realizations of S. $\alpha^n_{g=0,r=0}$ is used in all cases. The same 500 simulated parameter sets are used in all cases, only g is changed.

4.3.1.4 Performance of benchmark pricing function[23]

A SunGard finite differences based function will be used as our benchmark (compare section 4.3.2.1). One might ask oneself whether this function performs well. This short section aims to make it plausible that this is the case.

We take advantage of the fact that continuously monitored barrier options can be priced analytically. In fact, one of the SunGard functions does this for us. We examine these options and use the analytical function as benchmark this time. The absolute error is shown in Figure 13, for different numbers of steps, n_{steps}, in the finite differences based function. We see that using more steps yields a better value. We use 480 steps throughout this thesis, and we feel very comfortable using the finite differences based function with this number of steps as our benchmark pricing method since the *largest* absolute errors it gives rise to in Figure 13 are less than $1 \cdot 10^{-4}$.

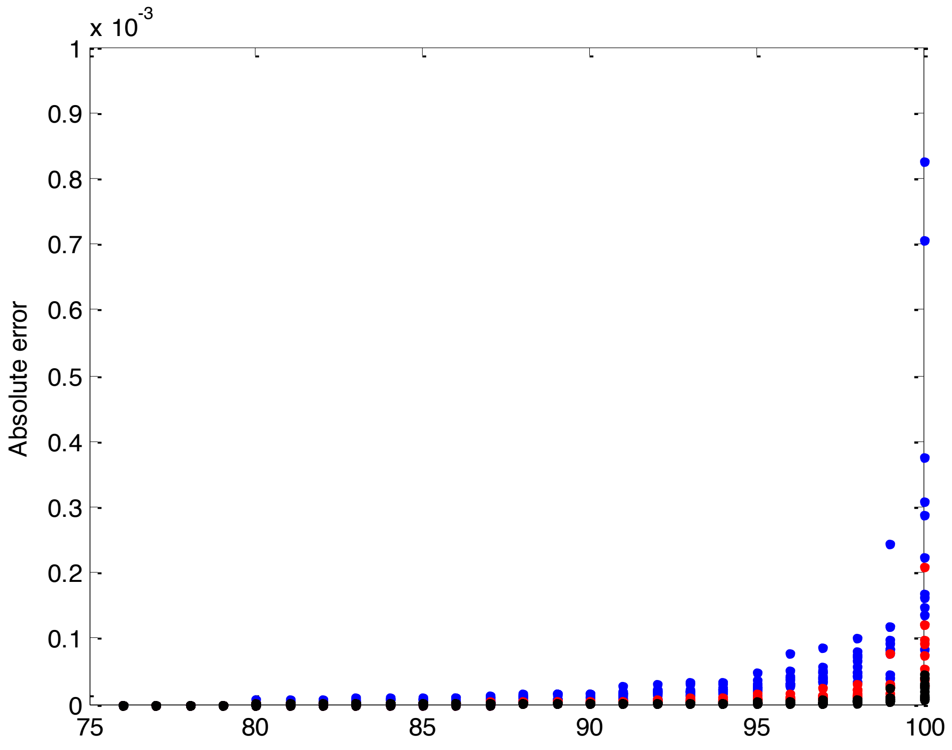

Figure 13: Blue dots correspond to 120 steps in the finite differences based function, red dots correspond to 240 steps, and black dots correspond to 480 steps. Each error series is sorted in ascending order, and each series is based on 100 simulated parameter sets of data with normally distributed values of the underlying. Only the largest errors are shown (see x-axis). The maximum absolute errors are $8 \cdot 10^{-4}$, $2 \cdot 10^{-4}$, and $5 \cdot 10^{-5}$ for the three series respectively.

[23] This section is relevant not only for problem one, but for the whole thesis.

4.3.2 Determination of optimal α

To determine optimal values of α we follow these steps:

1. Obtain option values corresponding to simulated data.
2. State optimization problems.
3. Choose optimization algorithm.
4. Determine optimal number of simulated values.
5. Determine optimal values of α.

The steps are described in turn below.

4.3.2.1 Option values corresponding to simulated data

To obtain option prices corresponding to the simulated data we need to call the SunGard option pricing functions, which are written in the C programming language. Our initial aim was to do this directly from Matlab. This proved far from trivial in practice, however. We use the following approach instead:

1. Write simulated values from Matlab to a .txt file using the Matlab function *dlmwrite*.
2. Read the values from the text file and store them in a matrix in C using a loop and the functions *fscanf* and *atof*.
3. Call the relevant SunGard option pricing functions with the simulated parameters as input values.
4. Write the parameter values along with the corresponding option prices to an .xls file using a loop and the function *fprintf*.
5. Read all data from the .xls file using the Matlab function *tdfread* and store it in a data structure in Matlab.

Of the steps outlined above, all except number three can be considered "support steps" which do not by themselves add much value (but which are nevertheless necessary). We will not describe these in detail. The third step will be described in the rest of this section, however.

The SunGard C functions that we call from our main C function are used to price barrier options analytically and numerically, respectively. The numerical pricing algorithm is the best one of the SunGard functions for pricing the discretely monitored variants, and the value it returns is considered the "correct" one for our purposes (it is our benchmark[24]). The numerical pricing method relies on a finite differences approach. The analytical function can either price a *continuously* monitored barrier option exactly, or approximate the price of a discretely monitored one using the 0.5826-approximation, possibly along with some internal SunGard improvements.

Due to secrecy we cannot give a more lengthy explanation of exactly how these SunGard option pricing formulas work. This does of course decrease the reader's possibilities to replicate all steps, and thereby his possibilities to replicate our results. The purpose of this thesis, however, is to mitigate the specific problems outlined in the problem definition. These problems exist in standard, unimproved pricing functions and SunGard's functions alike. It is therefore of less importance if we minimize the difference between option prices from SunGard's functions or between functions we have implemented ourselves. An important insight that this thesis provides is of course also the *method* for mitigating the identified problems. Anyone who has a fairly exact pricing method as well as an analytical formula that uses the 0.5826-adjustment will, if not be able to use our results right "off the shelf", at the very least be able to replicate our approach to obtain an α or equivalent suited to his needs.

[24] Note that this is the case throughout the whole thesis, and compare section 4.3.1.4 *Performance of benchmark pricing function*.

4.3.2.2 Optimization problems

We have two approaches for finding candidates for the optimal value of α. The first one is a vector based approach that minimizes the 1-norm of a vector of weighted differences between the true option value and our adjusted 0.5826-approximation, and finds only one α. The second one uses a loop and solves for the optimal α for each set of simulated parameters (one optimal α from m number of S, where m denotes the number of simulated parameter sets. We have used $m = 11$ throughout the thesis.).

The advantage of the vector based approach is that it gives us an optimal α which is really the best one we can find for our purposes (given our choices of norm and weight factor, which are discussed in section 4.3.2.2.1). This is beneficial since we, in our final recommendation, only want *one* optimal α based on each optimization.

The reason that the loop based approach is still useful is that we can analyze a lot of things when we get one optimal α for each parameter set (which we cannot with the vector based approach). We examine for example what "original" errors (from the plain 0.5826-approximation) correspond to different values of α (section 4.3.2.5.1) and make a sensitivity analysis of α (section 4.3.3.2). We also extract some statistics from the loop approach (section 4.3.2.5.2).

We need the delta in both approaches. It is approximated numerically for different parameter values as

$$\Delta(H_A) = C_{DO}^{0.5825}\big(T, \hat{S}, X, H, \sigma, r, g, \Delta t\big)/\varepsilon,$$

where $\hat{S} = H_A + \varepsilon$ and $\varepsilon = 1 \cdot 10^{-1}$. We have implicitly used the fact that $C_{DO}^{0.5825}\big(T, \tilde{S}, X, H, \sigma, r, g, \Delta t\big) = 0$ where $\tilde{S} = H_A$. As a service to the reader, we remind that the true value of the delta is given by

$$\Delta(H_A) = \frac{\partial}{\partial S} C_{DO}^{0.5826}(H_A).$$

The step size, ε, has been set to what has iteratively been concluded to be an appropriately large value to avoid numerical problems in general.

The optimization problems in the two approaches are stated in the following sections.

4.3.2.2.1 Minimization of norm of weighted errors

The idea in this approach is to minimize the 1-norm of a vector[25] of weighted errors between the true value and the approximated value using our proposed approximation, i.e. to obtain *one* optimal value of α for n simulated parameter sets. We solve the following problem numerically:

$$\min_{\alpha} \left[\left\| \boldsymbol{l} * \left(\boldsymbol{C}_{DO}^{FD} - \boldsymbol{V}^{eq} \right) \right\|_1 \right] \ over\ all\ \alpha \geq 0.$$

$\boldsymbol{l}$ will be defined in a moment (in the corresponding scalar form), and a motivation for using the 1-norm will also be given.

More explicitly (and in scalar form) the problem can be rewritten as

$$\min_{\alpha} \left[\sum_{i=1}^{n} \sum_{j=1}^{m} \left| l_{i,j} \cdot \left(C_{DO}^{FD}(T_i, S_{i,j}, X, H_i, \sigma_i, r_i, g_i, n_{monit,i}^{eq}, n_{steps}) - approximation_{i,j} \right) \right| \right] \ over\ all\ \alpha \geq 0,$$

with

[25] For this reason we sometimes refer to this approach as a "vector approach".

$$approximation_{i,j} = C_{DO}^{0.5826}(T_i, S_{i,j}, X, H_i, \sigma_i, r_i, g_i, n_{monit,i}^{eq}) + \Delta_i(H_{A,i}) \cdot \left(C(\hat{T}_i, S_{i,j}, \hat{X}_i, \sigma_i, \hat{r}_i, g_i) - max(S_{i,j} - \hat{X}_i, 0)\right),$$

where

$$\hat{r}_i = 0$$
$$\hat{T}_i = \alpha\Delta t_i$$
$$\hat{X}_i = H_{A,i} = H_i e^{-0.5826\sigma_i\sqrt{\Delta t_i}}$$
$$\Delta_i(H_{A,i}) = \frac{\partial}{\partial S} C_{DO}^{0.5826}(H_{A,i}).$$

n denotes the number of simulated parameter sets and m denotes the number of simulated values of the underlying in each set of simulated parameters. We have considered two cases – one with $l_{i,j} = 1$ for all i and j and one where $l_{i,j} = exp\left(-\frac{(S_{i,j}-H_{A,i})^2}{8(H_{A,i}\sigma_i\sqrt{\Delta t_i})^2}\right)$.

The behavior of the second weight factor can be seen in Figure 14.

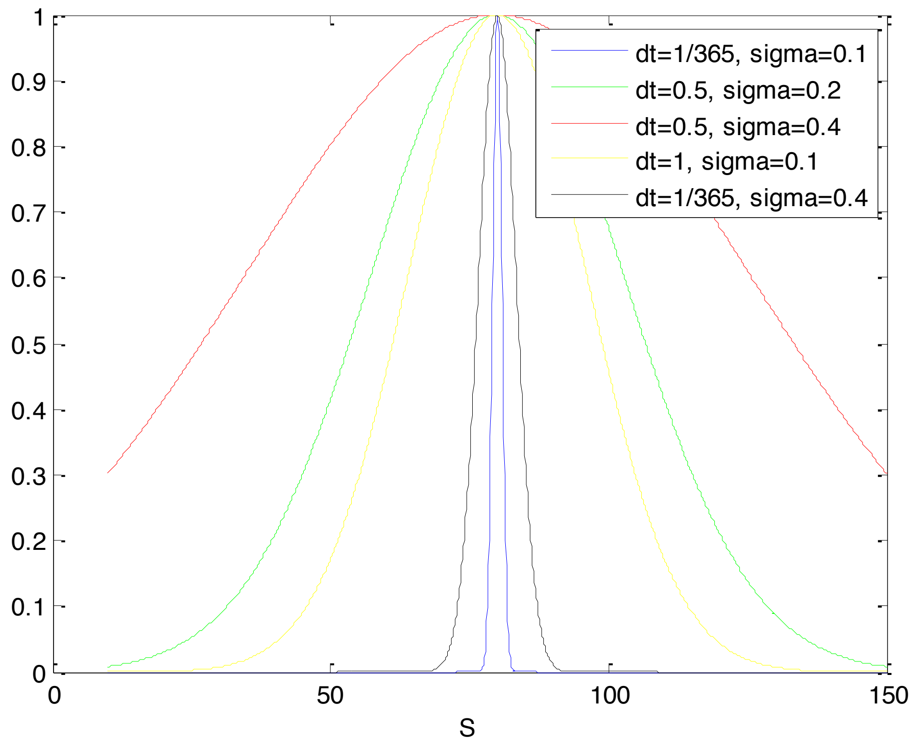

Figure 14: The weight factor for different values of Δt and σ. $H_A = 80$ for all curves. No other parameter values are relevant since the weight factor is independent of these.

With the different choices of $l_{i,j}$, α is optimized in "different ways". That is, the obtained optimal α will be (even) better for certain parameter values than for others.

Our first approach was to use $l_{i,j} = 1 \; \forall \; i,j$. This choice is not adapted to our purposes, however. That is because it does not take into account how close to, or far from, the adjusted barrier the errors occur (and as stated in section *1.2 Purpose* we focus on performance close to the (adjusted) barrier). We solved this problem by using the second alternative for $l_{i,j}$. As is evident from Figure 14, this factor is equal to one when $\frac{S_{i,j}}{H_{A,i}} = 1$, it is symmetric around $H_{A,i}$, and becomes gradually smaller (although it remains positive) when $S_{i,j}$ moves away from $H_{A,i}$. That means that errors for values of the underlying far away from the adjusted barrier (in terms of standard deviations, in a sense) will be considered less serious than errors close to it, which is exactly what we want to achieve. In the following we will only use this more sophisticated choice of $l_{i,j}$.

It is common to use the 2-norm instead of the 1-norm. Our reason for choosing the 1-norm is that we want errors of different magnitudes to be equally weighted (ceteris paribus). That would not have been the case for the 2-norm. If errors should at all be weighted differently, we think it makes more sense to exclusively weight them depending on the distance between the corresponding value of the underlying and the adjusted barrier (which is exactly what we do now).[26]

4.3.2.2.2 Minimization of absolute error for each simulated parameter set

The idea in this approach is to minimize the absolute error for each simulated parameter set, i.e. to obtain n optimal values of α. We solve the following (scalar) problem numerically in each step[27] (i.e. for each set of simulated parameters):

$$\min_{\alpha}[(C_{DO}^{FD} - V^{eq})] \ over \ all \ \alpha \geq 0.$$

The problem can be rewritten more explicitly as

$$\min_{\alpha}\left[\sum_{j=1}^{m}\left|C_{DO}^{FD}\left(T, S_j, X, H, \sigma, r, g, n_{monit}^{eq}, n_{steps}\right) - approximation_j\right|\right] \ over \ all \ \ \alpha \geq 0,$$

with

$$approximation_j = C_{DO}^{0.5826}\left(T, S_j, X, H, \sigma, r, g, n_{monit}^{eq}\right) + \Delta(H_A) \cdot \left(C\left(\hat{T}, S_j, \hat{X}, \sigma, \hat{r}, g\right) - max\left(S_j - \hat{X}, 0\right)\right)$$

where

$$\hat{r} = 0$$
$$\hat{T} = \alpha\Delta t$$
$$\hat{X} = H_A = He^{-0.5826\sigma\sqrt{\Delta t}}$$
$$\Delta(H_A) = \frac{\partial}{\partial S} C_{DO}^{0.5826}(H_A).$$

4.3.2.3 Optimization algorithm[28]

A problem in the first approach (described in section 4.3.2.2.1) is that there may be multiple local minima. Thus, when we started out with a naïve approach based on the Matlab local optimization algorithm *fmincon*, we quickly ran into trouble, particularly for poorly chosen initial points, and particularly in the more complicated optimization problems described in section 4.3.4. The optimizer tended to find a local minimum (but sometimes not the global one).

To overcome this problem we started to read up on, and try, various optimizers that are supposed to be better at finding global minima. All methods tested have been Matlab functions.

[26] The same motivation applies for using the 1-norm when solving the second problem in section 5, with the exception that we do not even weight depending on distance there.
[27] This is implemented with a loop in Matlab. This is why we sometimes refer to this approach as a "loop approach".
[28] This section is relevant not only for this first problem, but for problem two as well.

GlobalSearch (with and without a multi-start feature) has been tested, but did not particularly improve the results obtained with *fmincon*. We then moved on to *patternsearch* (via a *Simulated annealing* solver, which was not successful). The pattern search method, with default settings, produces markedly better results. The values of the objective function is improved for most of the functions we minimize, and it shows no tendency of getting stuck in local minima.

"The pattern search method handles optimization problems with nonlinear, linear, and bound constraints, and does not require functions to be differentiable or continuous [10]." The idea of pattern search is to use a mesh, the size of which is gradually adapted until an optimum is found [11]. A few (default) or all (optional) mesh points are evaluated in each iteration. As default, there is no so called search step but only a poll step. If a search step is included, as we have done, a separate method is used first in the search step. If this method finds a better objective value, this is accepted and the poll step is skipped. If it does not, the chosen poll method comes in and hopefully finds a better objective value instead. Complete search and poll ensure that each point in the mesh, as opposed to only a few, is evaluated in each iteration [12]. Both including a search step [13] and enabling these features increases the probability of finding a global, rather than a local, minimum. We have consequently done both. We have also for instance (drastically) increased the maximum number of function evaluations and iterations, and decreased the mesh size tolerance.

For the search step we use the generalized pattern search method *GPSPositiveBasis2N* and for the poll step we use the mesh adaptive search method *MADSPositiveBasis2N*.

"The default pattern, GPS Positive basis 2N (GPSPositiveBasis2N), consists of the following 2*N* vectors [12], where *N* is the number of independent variables for the objective function.

[1 0 0...0][0 1 0...0] ...[0 0 0...1][–1 0 0...0][0 –1 0...0][0 0 0...–1]."

"The MADS Positive basis 2N pattern (MADSPositiveBasis2N) consists of 2*N* randomly generated vectors, where *N* is the number of independent variables for the objective function. This is done by randomly generating *N* vectors which form a linearly independent set, then using this first set and the negative of this set gives 2*N* vectors [12]."

We have also experimented with *Genetic Algorithm* and *Latin hypercube* in the search step. The performance of these has not been as good, however.

There is a potential drawback with pattern search in the way the mesh size is adapted in each step. It is decreased if a better point is not found in a certain iteration, and increased otherwise [11]. If we happen to start exactly in a local minimum, and there are no better objective values "within reach", there is a risk that we get stuck. This is not considered a large problem in our case, however, since we have a pretty good idea of where the minimum is from using other algorithms, and since we have tried with different starting points. Furthermore, we have verified that pattern search finds a better optimum than the other tested algorithms in several cases relevant to our work. It might be an issue worth keeping in mind in the extensive test we perform in section 4.3.4.1, however.

The optimization in the second approach (described in section 4.3.2.2.2) is less advanced since there are no vectors involved. We minimize a function that only depends on one variable. For this, the Matlab (local) optimizer *fmincon* has been used. We have had no indications that this algorithm is not up to the task.

4.3.2.4 Optimal number of simulated parameter sets

There are several factors to consider when deciding on an “optimal” number of simulated parameter values. In terms of accuracy, it is of course always best to have as many simulated values as possible. To have time to actually analyze the simulated data (since certain calculations take a lot of computation time), however, it is tempting to simulate as few values as possible.[29]

We have examined the extent to which the value of α is affected by different numbers of simulations in the vector approach, with $g = 0, r = 0$. The results are visualized in Figure 15.

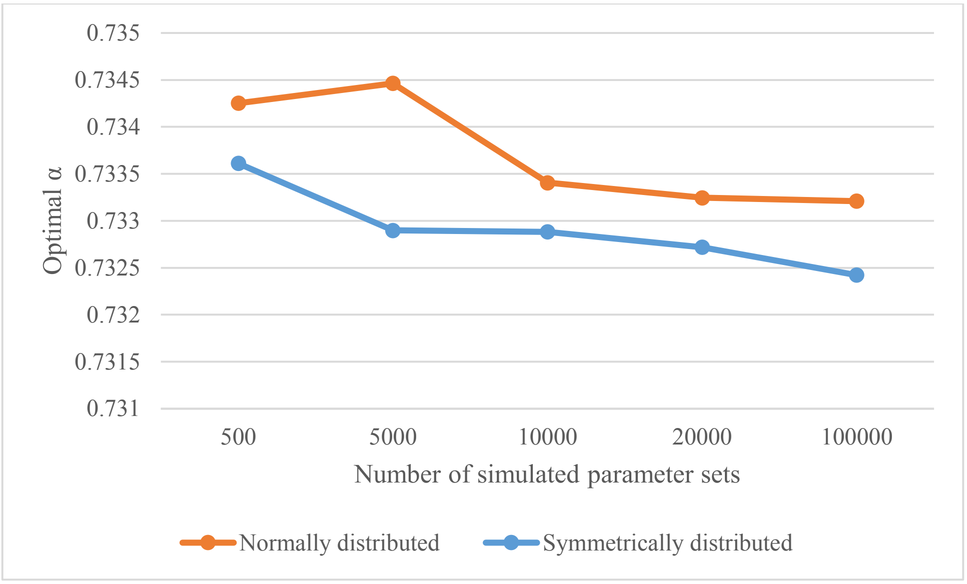

Figure 15: Optimal value of α for different numbers of simulated parameter sets, calculated using the first optimization approach with our own weight factor.

It can be concluded that, although the optimal value of α changes when the number of simulations is increased, the difference is not particularly large. The value of the optimal α is only 0.142 and 0.162 percent larger for 500 than for 100,000 simulations for the data with normally and symmetrically distributed S, respectively. The corresponding percentages for 20,000 simulations are 0.005 and 0.040. Although it seems likely that the optimal α may converge to an even smaller value, we have to set *some* limit for the accuracy. Based on this analysis we feel comfortable to settle for 20,000 simulations in the vector approach. This number will be used in all the main tests aimed at finding optimal values of α. In the loop approach we will use different numbers of simulations from time to time since we consider this number less important (compare section 4.3.2.5).

[29] The simulation of parameter values is in fact done in almost no time at all. It is the option pricing and the loop approach that consume the most time.

4.3.2.5 Optimal α

In this section we set out to find optimal values of α. We first examine how the original errors (true value versus 0.5826-approximation) are distributed in relation to values of α obtained with the second approach (the loop). We then present the main results – the solutions of the optimization problems stated in section 4.3.2.2.

4.3.2.5.1 Distribution of errors in relation to α[30]

It can be informative to examine what magnitude of errors correspond to different values of α. A good situation would be if it turned out that the largest errors correspond to fairly typical values of α, because in that case we can feel comfortable that choosing such a value of α will with a high probability mitigate many of the largest errors.

In Figure 16 (where the α's are calculated from symmetrically distributed S) you can see that the abnormally small values of α correspond to relatively few large errors. The same goes for Figure 17 (where the α's are calculated from normally distributed S). At the same time, however, it must be admitted that the tendency is not overly clear.

We cannot conclude that abnormal values of α can be *ignored* based on this, but we can see that the *largest* errors are at least typically not obtained for these values. This means that we can feel confident that choosing a typical value of α will mitigate many large errors.

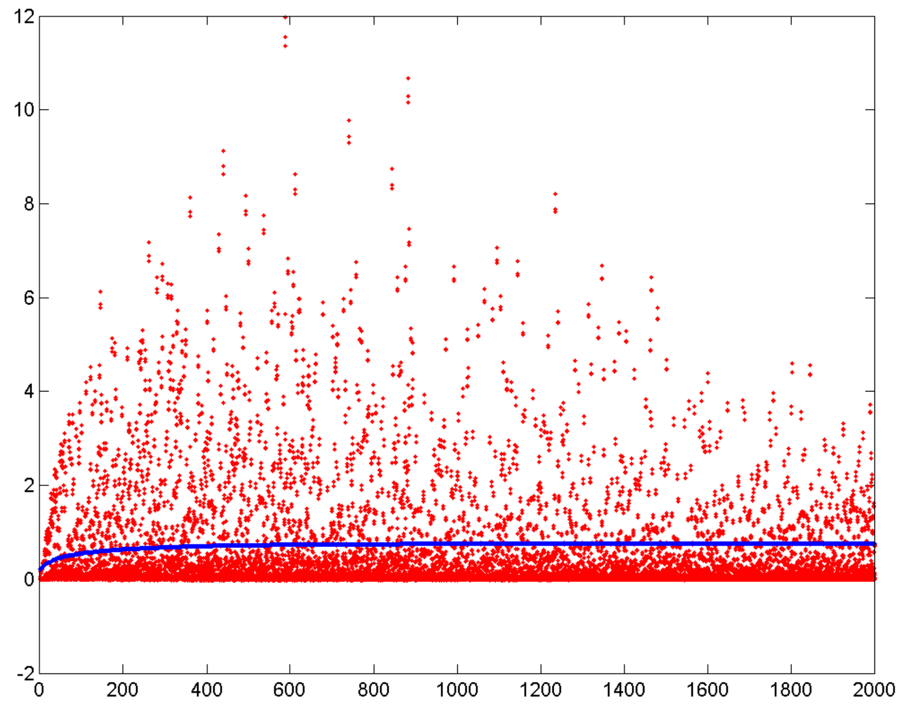

Figure 16: Calculated sorted α's (blue) and corresponding errors between the true value and the plain 0.5826-approximation (red) for all symmetrically distributed S (11 different).

[30] The analysis in this section has been carried out for data sets with 2,000 simulations.

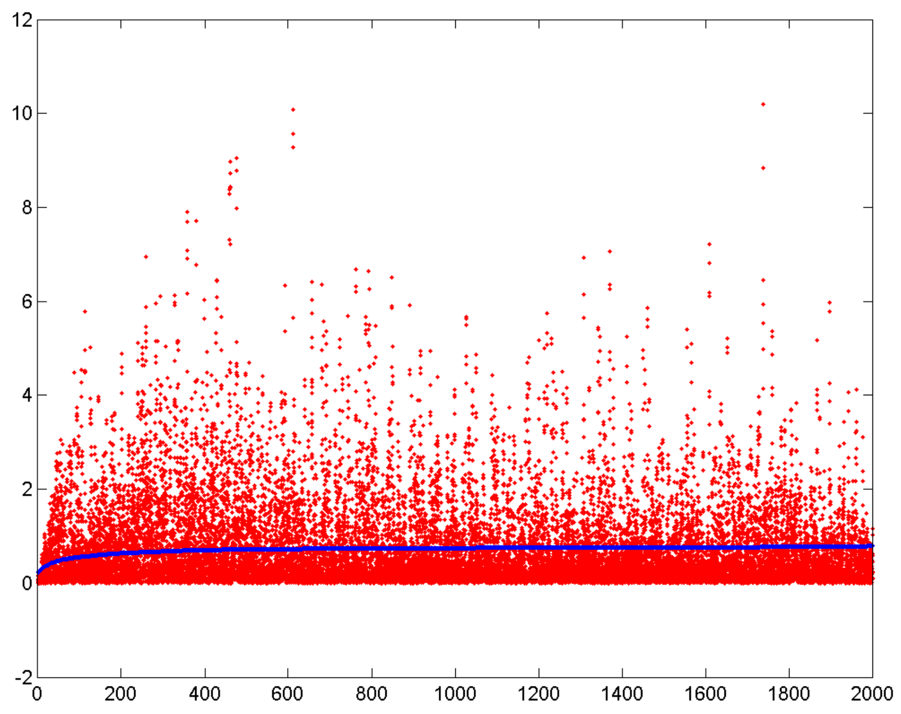

Figure 17: Calculated sorted α's (blue) and corresponding errors between the true value and the plain 0.5826-approximation (red) for all normally distributed S (11 different).

4.3.2.5.2 Results from solving the optimization problems

The results from the two approaches described in section 4.3.2.2 are different in the sense that the second approach (minimization for each simulated parameter set) returns a huge number of α-values whereas the first one (minimization of norm of weighted errors) yields only one. We are confident that the vector based (first) approach produces the most accurate α (since all simulated parameter sets are here considered at once), but the output from the loop (second approach) is useful for obtaining some simple statistical measures, for analyzing relations between other parameters and the values of α, and as a reality check.

When rounded to three decimals, the first approach with both normally and symmetrically distributed values of S yields an optimal value of α of $\alpha^{n}_{g=0,r=0} = 0.733$.

With simulated g and r different from zero we get a smaller value, and larger differences between the results with the different distributions of S. The data with a symmetrically distributed S give an optimal $\alpha = 0.599$ and the data with normally distributed S give $\alpha = 0.638$. We use the average of the two as optimal value, i.e. $\alpha^{n}_{g\neq 0,r\neq 0} = 0.618$.

It can be informative to provide some statistics from the second approach[31]. The average value of α with g and r equal to zero is 0.722, and the corresponding value for g and r different from zero is 0.662. The median values from the two cases are 0.749 and 0.705, respectively. It can also be useful to look at the sorted obtained values of α as a "sanity check" for the first approach, see Figure 18 and Figure 19.

[31] The statistics are calculated on 10,000 simulated parameter sets, with normally distributed values of S.

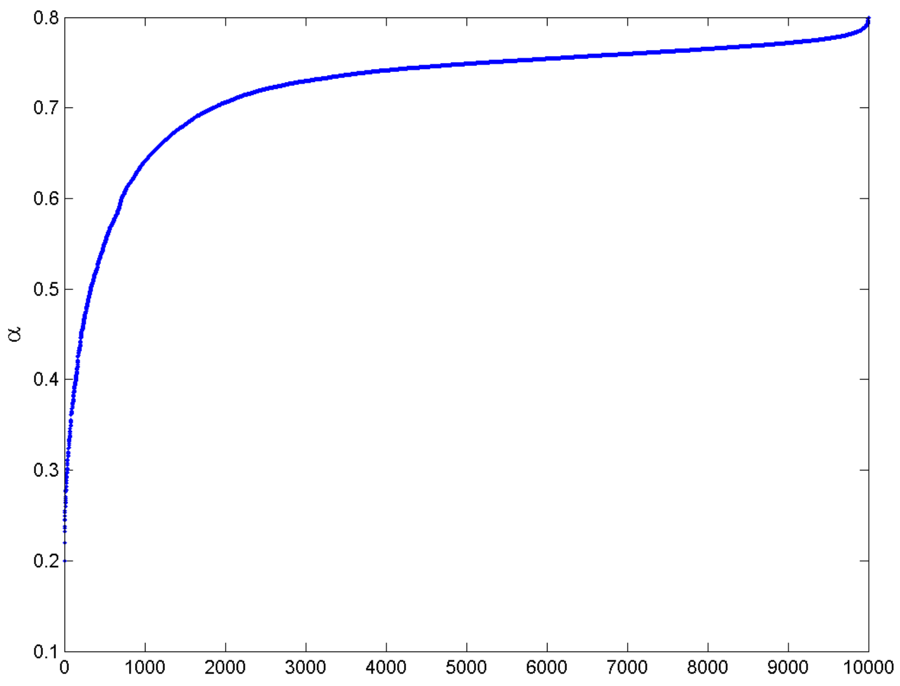

Figure 18: The values of α obtained with the loop approach with normally distributed S, sorted in ascending order.

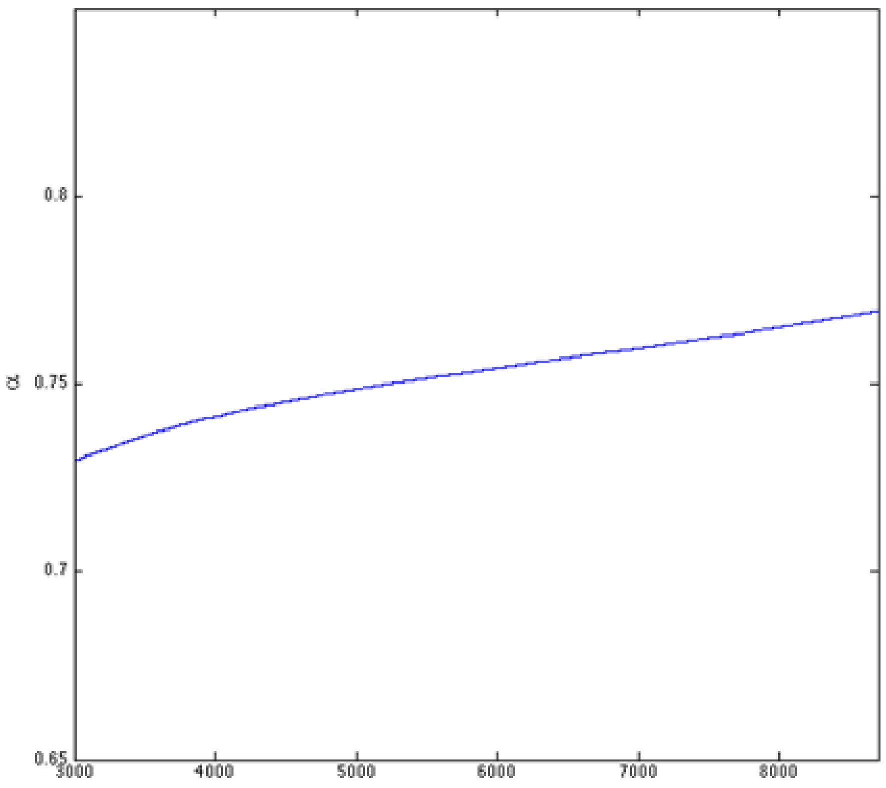

Figure 19: The values of α obtained with the loop approach with normally distributed S, sorted in ascending order and zoomed in on the flatter part. That is, this is a zoomed in version of Figure 18.

We can conclude that the statistics given from the second approach are at least not completely different from the optimal values obtained with the first approach. Because of this and the fact that we think that the first approach is more adapted to meet our goals, we do not see any reason not to stick with the optimal values of α obtained with the first approach.

The optimal values of the coefficient α in this first problem are thus

$$\alpha^n_{g=0,r=0} = 0.733,$$
$$\alpha^n_{g\neq0,r\neq0} = 0.618.$$

4.3.3 Test of optimal α

In this section we test the suggested adjustment with the optimal values of α obtained in section 4.3.2.5 on simulated data that were not used to obtain these. We also make a sensitivity analysis of α.

4.3.3.1 Accuracy of resulting approximation

We test $\alpha^n_{g=0,r=0}$ both on data with $g = 0, r = 0$ and on data with $g \neq 0, r \neq 0$. The same goes for $\alpha^n_{g\neq0,r\neq0}$. The improvement is analyzed and we examine how well the adjustment works for different Δt and n^{eq}_{monit}. For each data type we have simulated 100,000 parameter sets with normally distributed values of S. 20,000 of these have been used to obtain the optimal value of α in each case, whereas the remaining 80,000 are used for the testing[32]. In this way we make sure that the optimal value is not tested on data which have been used to obtain it.

4.3.3.1.1 Analysis with g and r equal to zero (optimal α based on $g = 0$ and $r = 0$)

In this section we test $\alpha^n_{g=0,r=0} = 0.733$ on data with $g = 0, r = 0$.

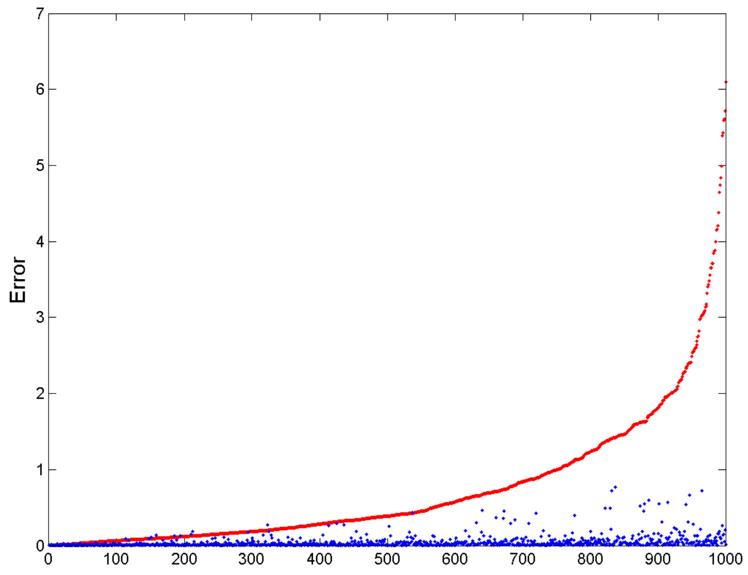

Figure 20: Sorted absolute differences between the true value and 0.5826-approximation (red) and respective absolute differences between the true value and our adjusted 0.5826-approximation (blue). The plot was obtained using 1,000 simulated parameter sets.

[32] Since there are 11 different values of S in each parameter set, this corresponds to 880,000 individual tests.

In Figure 20 you can see that using the adjustment significantly improves the pricing of the barrier option, especially for large original errors. Our adjustment improves[33] the approximation of the price of the barrier option in 862,935 cases out of 880,000 (98.1%).

The mean absolute error we get from only using the 0.5826-approximation is 0.737 and the median 0.398. When using the new adjustment we get a mean absolute error of 0.0451 and a median of 0.0183. The new approximated values of the barrier options differ from the true values by on average 75.2% less than the ones calculated without our adjustment (that is, the mean improvement is 75.2%). The median improvement is 95.0%.

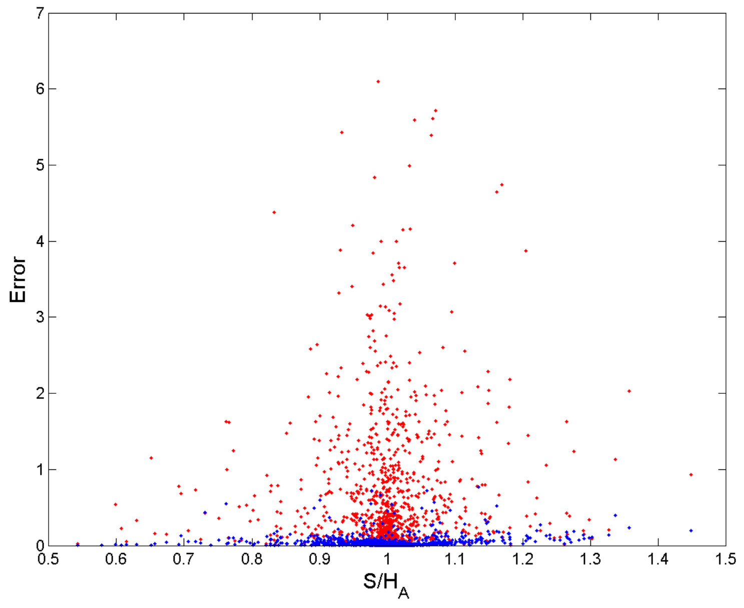

Figure 21: Absolute differences between the true value and the 0.5826-approximation (red) and absolute differences between the true value and our adjusted 0.5826-approximation (blue) plotted against S/H_A. The plot was obtained using 1,000 simulated parameter sets.

We want our adjustment to work especially well when the underlying is close to the adjusted barrier, and as you can see in Figure 21 we get no large errors close to the adjusted barrier (as opposed to when using the plain 0.5826-approximation).

Our adjustment is almost always improving the result compared to just using the 0.5826-approximation. In Figure 22 you can see the sorted differences between the absolute normalized errors from the plain 0.5826-approximation and the corresponding errors using the 0.5826-approximation with the adjustment based on $\alpha_{g=0,r=0}^{n}$. The normalization of errors corresponding to each simulated parameter set is done by dividing them by the absolute error we get from just using the plain 0.5826-approximation when the underlying is equal to the adjusted barrier ($S = H_A$) for that same parameter set. The rationale is that the error at the adjusted barrier should be a good proxy for the maximum absolute error for each parameter set. As you can see, there are only a few adjustments that give worse results. Specifically, there is an improvement in 98.1% of the cases.

[33] We are referring to non-strict improvements, that is the new error is less than or equal to the old error. This is the case in all of section 4.3.3.

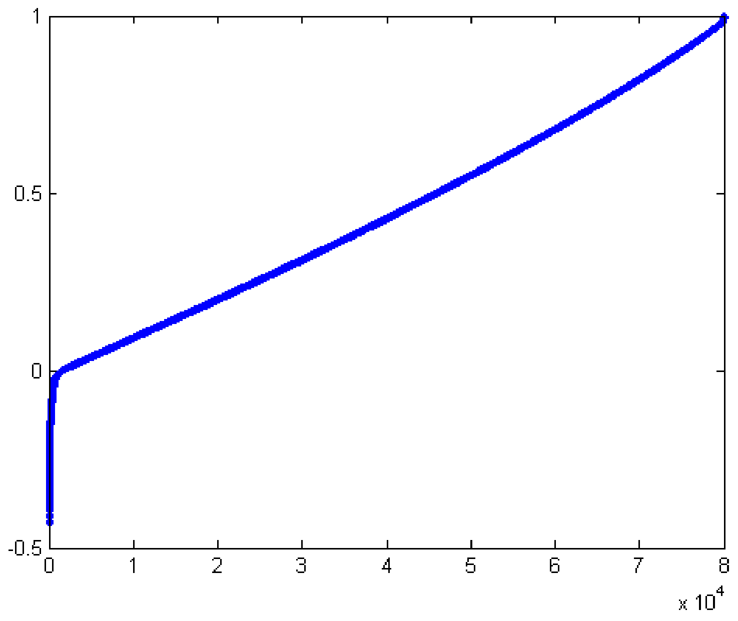

Figure 22: The sorted normalized differences between the absolute differences between the true value and the 0.5826-approximation and the absolute differences between the true value and our adjusted 0.5826-approximation. The plot was obtained using 80,000 simulated parameter sets.

In Figure 23 and the corresponding zoomed in version, Figure 24, it can be seen that the performance of our proposed approximation gets worse when Δt increases, all else equal, but that it is still in general better than using the plain 0.5826-approximation.

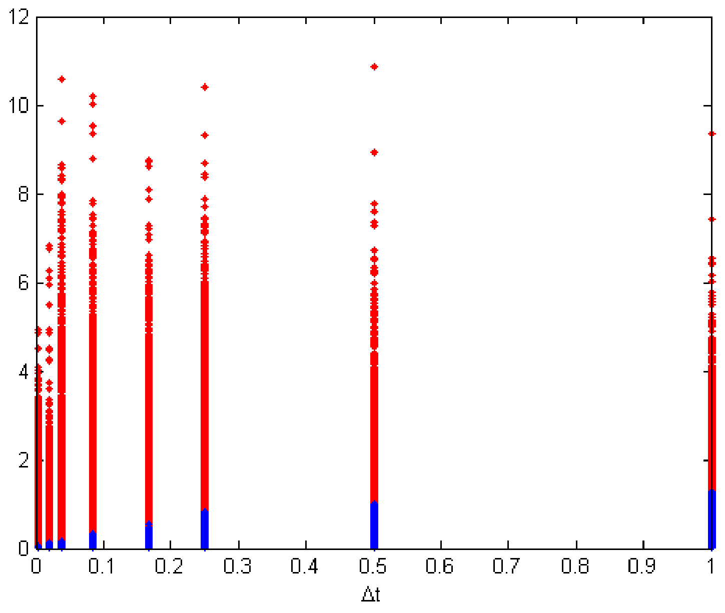

Figure 23: The absolute differences between the true value and the 0.5826-approximation (red) and the absolute differences between the true value and our adjusted 0.5826-approximation (blue) plotted against Δt. The plot was obtained using 80,000 simulated parameter sets.

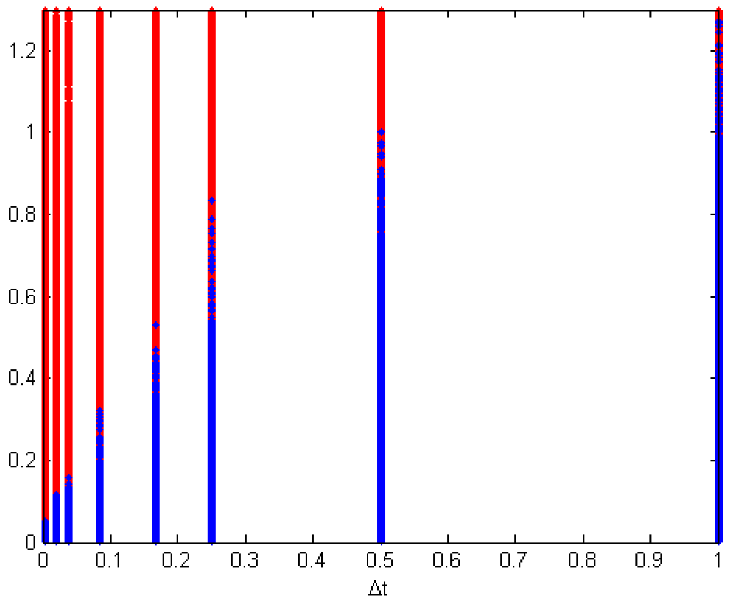

Figure 24: Zoomed in version of Figure 23.

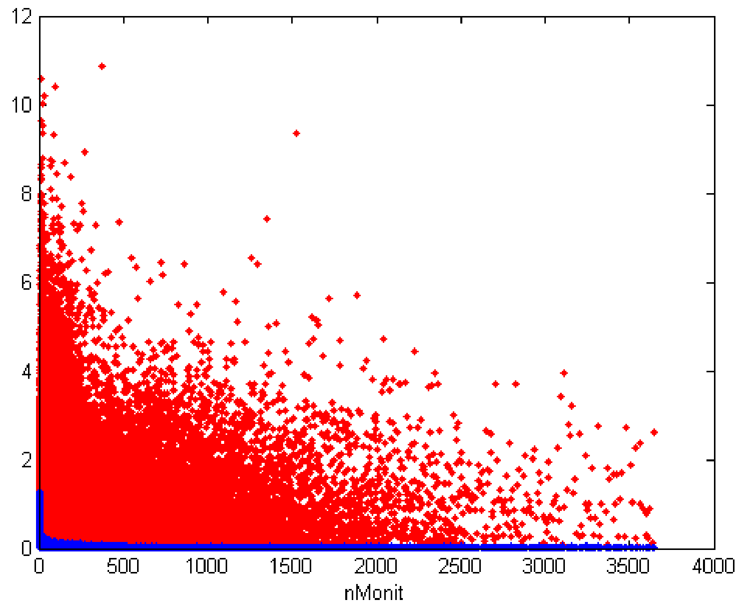

Figure 25: The absolute differences between the true value and the 0.5826-approximation (red) and the absolute differences between the true value and our adjusted 0.5826-approximation (blue) plotted against n_{monit}^{eq}. The plot was obtained using 80,000 simulated parameter sets.

Figure 25 shows that our adjustment significantly improves the price approximation overall. It does, however, perform slightly worse for small values of n_{monit}^{eq} (< 3) than for large values, which can be seen in Figure 26.

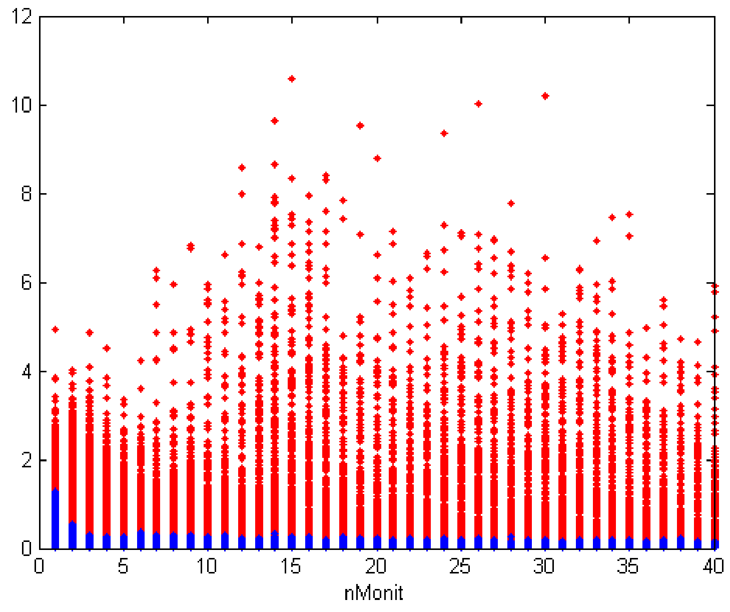

Figure 26: Zoomed in version of Figure 25.

Finally, out of curiosity, does the performance of the adjustment in the example case from section 4.2.1 improve when the optimal $\alpha^n_{g=0,r=0} = 0.733$ is used? The answer is clearly yes, which can be seen by comparing Figure 27 to Figure 8.

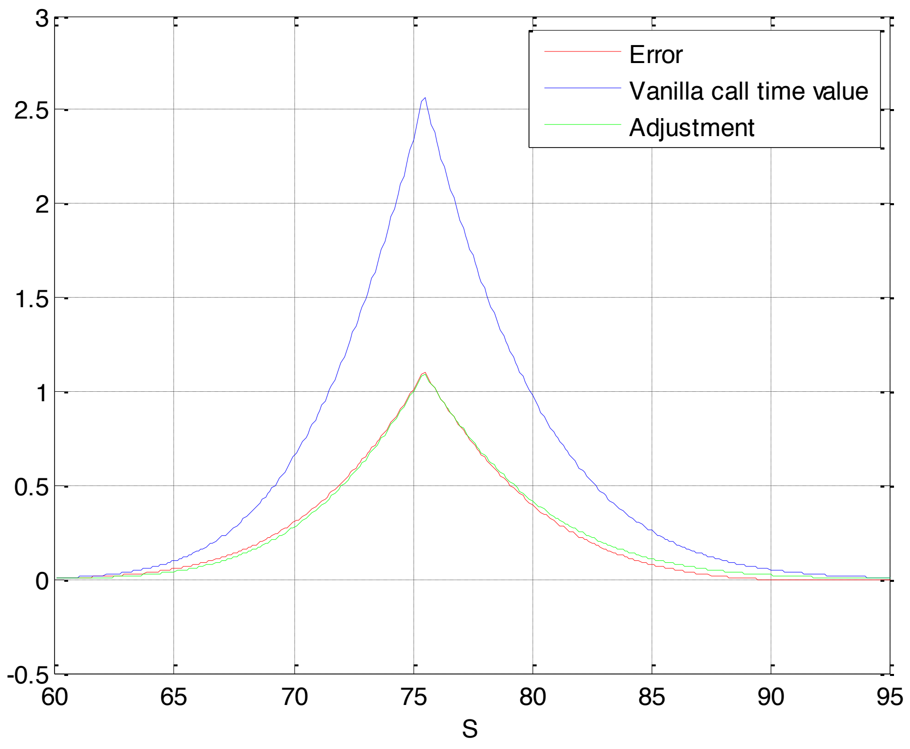

Figure 27: T The red curve shows the error between the benchmark ("true") value and the 0.5826-approximation. The blue curve depicts the time value of the appropriately chosen vanilla call option. The green curve shows the hypothesized adjustment (4.2), which would in an ideal case be exactly equal to the error. Parameter values: $X = 100, g = 0, r = 0, \sigma = 0.1, T = 10, n^{eq}_{monit} = 10, H = 80$. This gives $H_A \approx 75.5$. We use $\alpha^n_{g=0,r=0} = 0.733$. Compare Figure 8.

4.3.3.1.2 Analysis with simulated g and r (optimal α based on $g = 0$ and $r = 0$)

In this section we test $\alpha^n_{g=0,r=0} = 0.733$ on data with $g \neq 0, r \neq 0$.

In Figure 28 you can see that using the adjustment significantly improves the pricing of the barrier option in most cases, especially for large original errors (using the plain 0.5826-approximation). In 821,796 cases out of 880,000 (93.39%) our adjustment gives a more accurate result than just using the 0.5826-approximation.

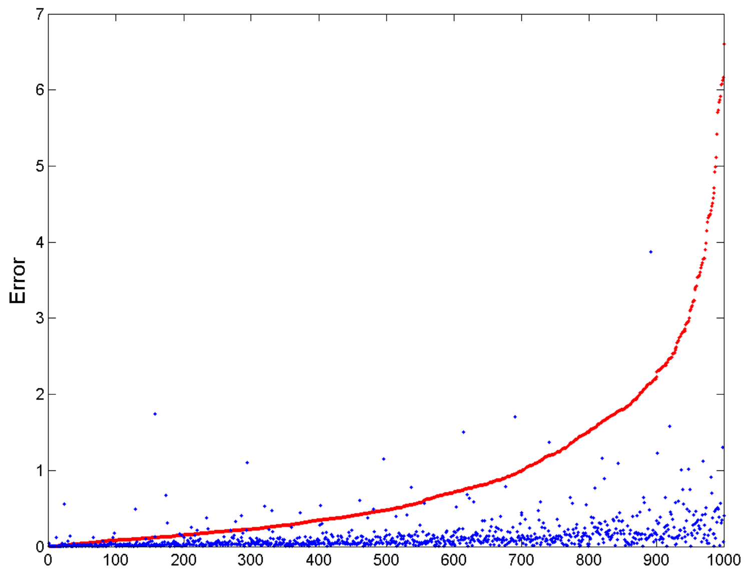

Figure 28: Sorted absolute differences between the true value and the 0.5826-approximation (red) and respective absolute differences between the true value and our adjusted 0.5826-approximation (blue). The plot was obtained using 1,000 simulated parameter sets.

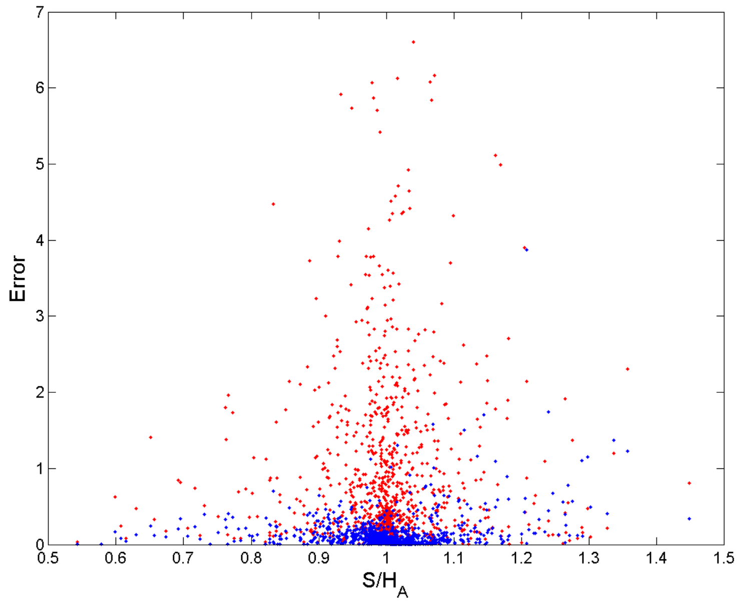

Figure 29: Absolute differences between the true value and the 0.5826-approximation (red) and the absolute differences between the true value and our adjusted 0.5826-approximation plotted against S/H_A. The plot was obtained using 1,000 simulated parameter sets.

The mean absolute error we get from only using the 0.5826-approximation is 0.890 and the median 0.490. When using the new adjustment we get a mean absolute error of 0.153 and a median of 0.0336. The mean improvement is 7.47% and the median improvement is 91.4%.

As mentioned before, the 0.5826-approximation performs the worst when the underlying is close to the adjusted barrier and we therefore want our adjustment to render an especially large improvement here. You can see in Figure 29 that this is in general achieved.

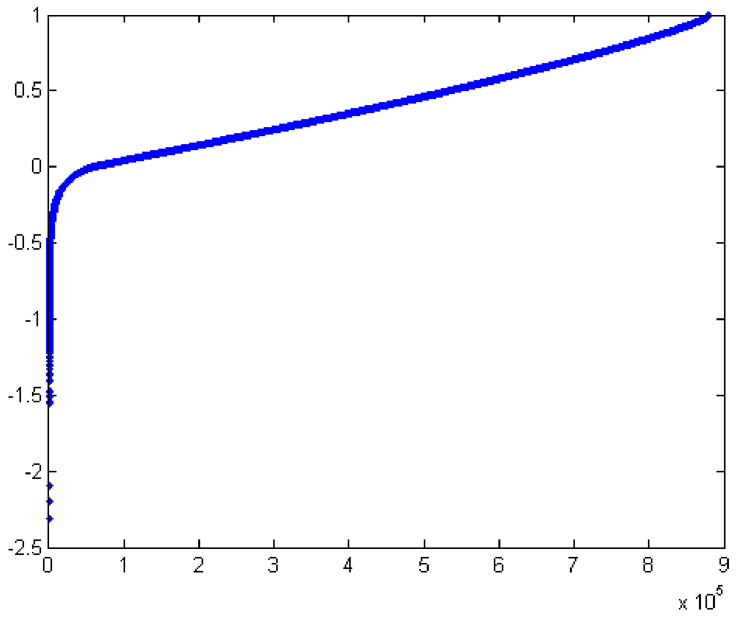

Figure 30: Sorted normalized differences between the absolute differences between the true value and the 0.5826-approximation and the absolute differences between the true value and our adjusted 0.5826-approximation. The plot was obtained using 80,000 simulated parameter sets.

Once again (compare section 4.3.3.1.1) we use the absolute error between the true value and the 0.5826-approximated price at the adjusted barriers as the “maximum errors” to normalize the absolute errors for each parameter set. In Figure 30 you can see the sorted differences between the absolute normalized errors from the plain 0.5826-approximation and the corresponding errors using the 0.5826-approximation with the adjustment based on $\alpha^{n}_{g=0,r=0}$. For 93.4% of the simulated parameter sets we get an improvement.

In Figure 31 it can be seen that the performance of our proposed approximation also in this case (compare section 4.3.3.1.1) gets worse when Δt increases, all else equal, but that it is in general better than using the plain 0.5826-approximation. In this case, when $\Delta t = 1$, the adjustment even seems to give some errors that are worse than just using the 0.5826-approximation.

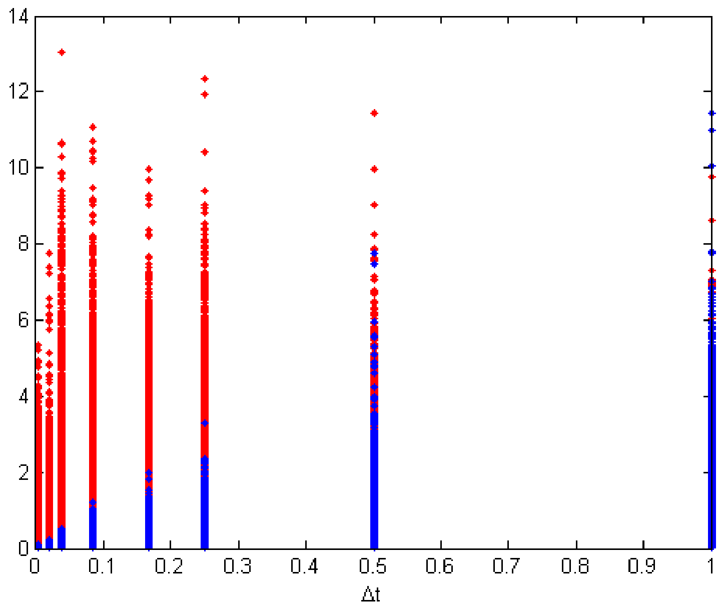

Figure 31: The absolute differences between the true value and the 0.5826-approximation (red) and the absolute differences between the true value and our adjusted 0.5826-approximation (blue) plotted against Δt*. The plot was obtained using 80,000 simulated parameter sets.*

In Figure 32 you can see that our adjustment seems to perform worse when the number of monitoring instants (n_{monit}^{eq}) is small (than when it is a bit larger).

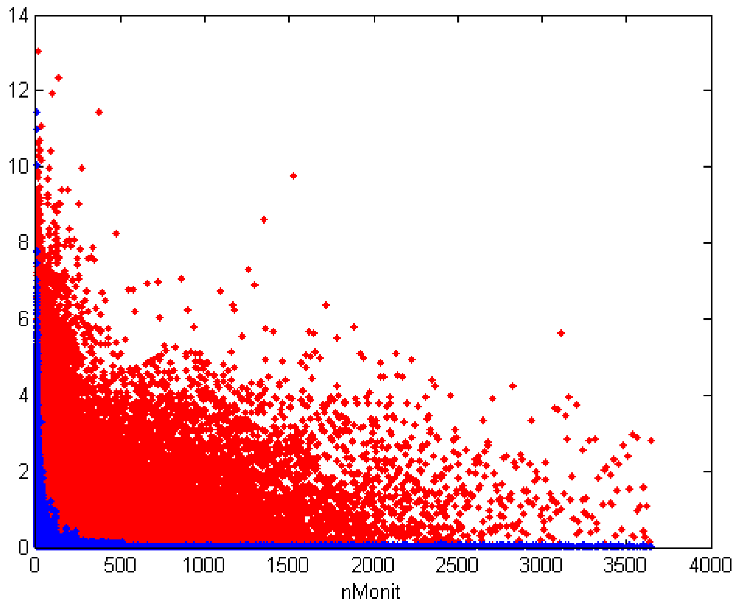

Figure 32: The absolute differences between the true value and the 0.5826-approximation (red) and the differences between the true value and our adjusted 0.5826-approximation (blue) plotted against n_{monit}^{eq}*. The plot was obtained using 80,000 simulated parameter sets.*

Our adjustment seems to work well when $n_{monit}^{eq} > 9$. This is easily seen in the zoomed in version of the plot, Figure 33.

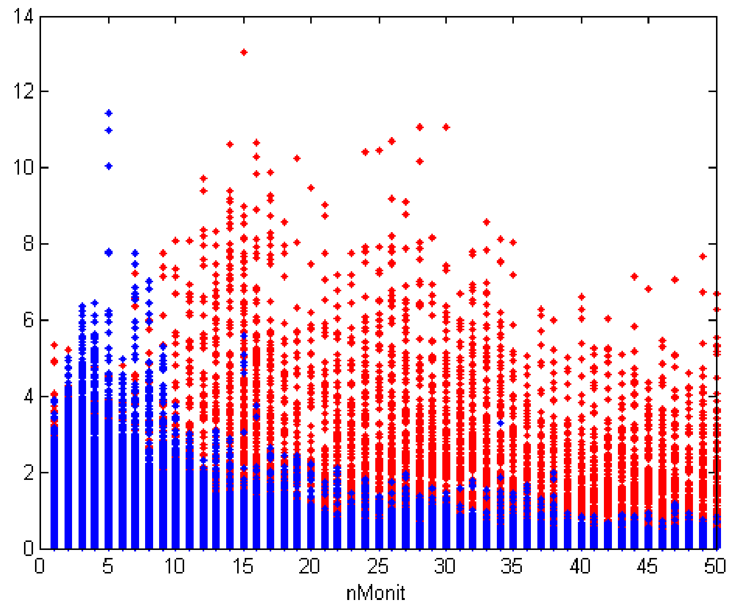

Figure 33: Zoomed in version of Figure 32.

4.3.3.1.3 Analysis with simulated g and r (optimal α based on simulated g and r)

In this section we test $\alpha^{n}_{g\neq 0,r\neq 0} = 0.618$ on data with $g \neq 0, r \neq 0$.

In Figure 34 you can see that using our approximation in general drastically improves the pricing also in this case. Now we get an improvement with our adjustment in 843,302 cases out of 880,000 (95.8%).

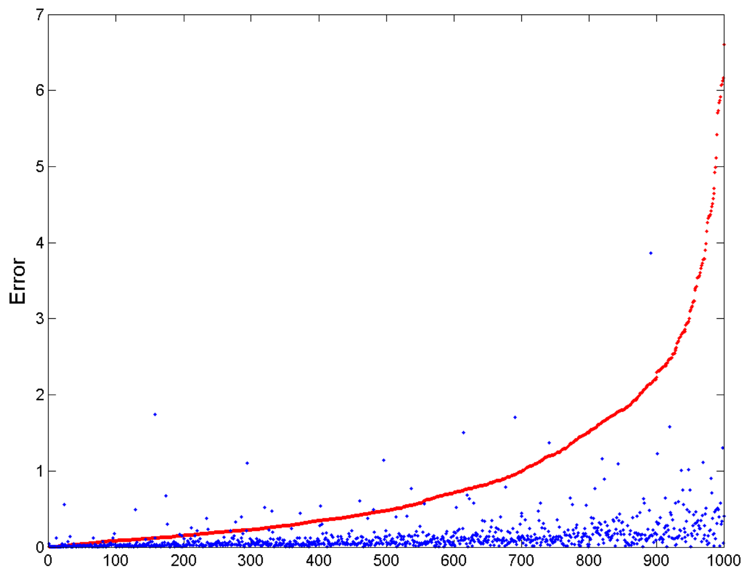

Figure 34: Sorted absolute differences between the true value and the 0.5826-approximation (red) and respective absolute differences between the true value and our adjusted 0.5826-approximation (blue). The plot was obtained using 1,000 simulated parameter sets.

When using the adjustment (with $\alpha^{n}_{g\neq 0,r\neq 0}$) on the 0.5826-approximation we get a mean absolute error of 0.132 (without adjustment: 0.900) and a median of 0.0656 (without adjustment: 0.490). The mean improvement is 23.5% and the median improvement is 87.1%.

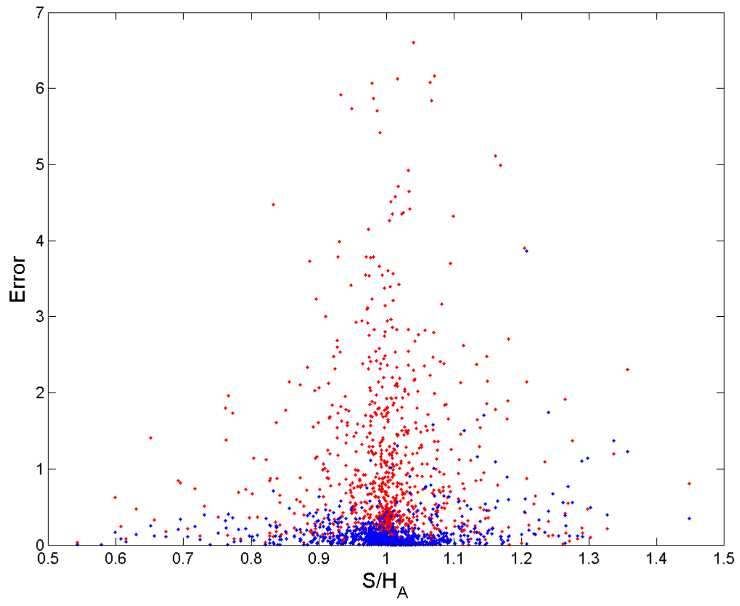

Figure 35: The absolute differences between the true value and the 0.5826-approximation (red) and the absolute differences between the true value and our adjusted 0.5826-approximation (blue) plotted against S/H_A. The plot was obtained using 1,000 simulated parameter sets.

We still see that our adjustment results in substantial improvements when the underlying is close to the adjusted barrier, Figure 35.

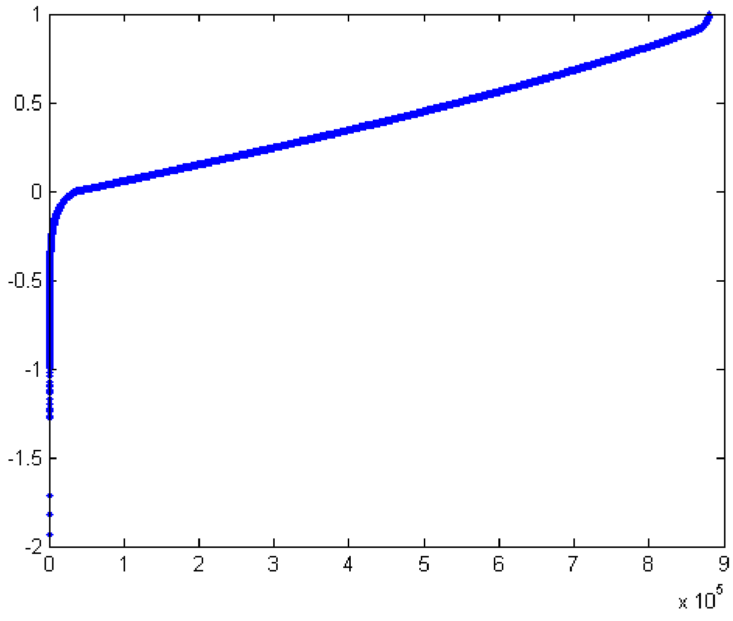

Figure 36: The sorted normalized differences between the absolute differences between the true value and the 0.5826-approximation and the absolute differences between the true value and our adjusted 0.5826-approximation. The plot was obtained using 80,000 simulated parameter sets.

Our improvement based on $\alpha^{n}_{g\neq 0, r\neq 0}$ works well for the data with $g \neq 0$ and $r \neq 0$. In Figure 36 you can see the sorted difference between the normalized absolute errors we get by using the plain 0.5826-approximation and the corresponding absolute errors obtained by using the 0.5826-approximation with our adjustment. Once again the "maximum absolute error" (which is used for the normalization) is calculated as the absolute difference between the true value and the 0.5826-approximation, both calculated with the underlying equal to the adjusted barrier. We get improvements for 95.8% of the simulated parameter sets.

As expected, when the test data contains simulated g and r different from zero, it seems to be better to use the corresponding optimal α, $\alpha^n_{g\neq0,r\neq0}$, in our adjustment than to use $\alpha^n_{g=0,r=0}$. We see for instance that we get better results for large Δt when using the $\alpha^n_{g\neq0,r\neq0}$ (compare Figure 31 and Figure 37). Performance is also improved when n^{eq}_{monit} is small (compare Figure 32 and Figure 38).

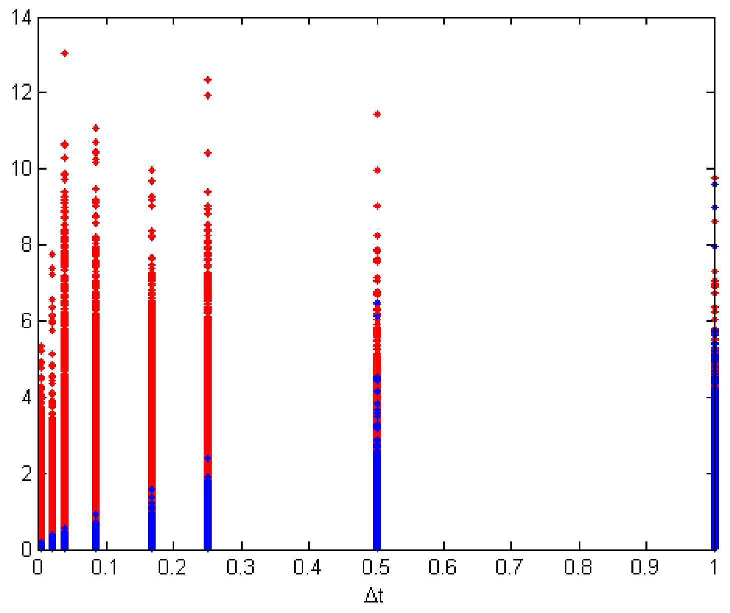

Figure 37: The absolute differences between the true value and the 0.5826-approximation (red) and the absolute differences between the true value and our adjusted 0.5826-approximation (blue) plotted against Δt. The plot was obtained using 80,000 simulated parameter sets.

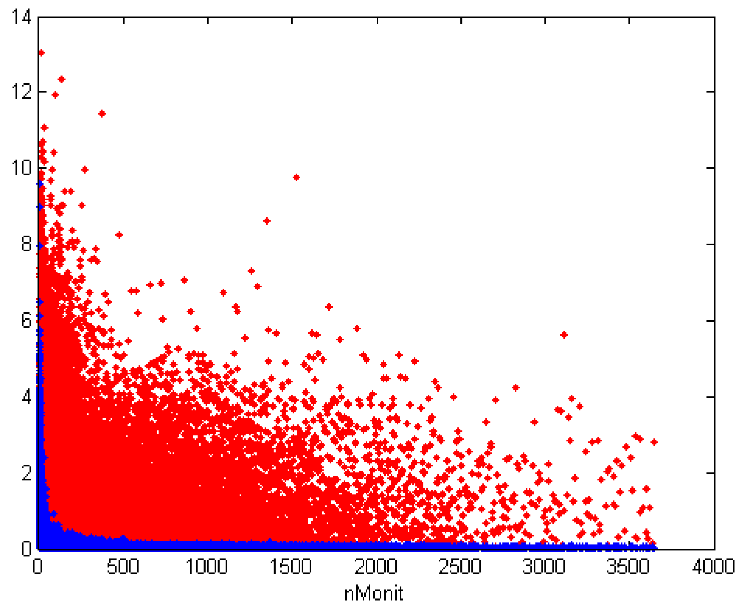

Figure 38: The absolute differences between the true value and the 0.5826-approximation (red) and the absolute differences between the true value and our adjusted 0.5826-approximation (blue) plotted against n^{eq}_{monit}. The plot was obtained using 80,000 simulated parameter sets.

Our improvement still seems to be great when $n^{eq}_{monit} > 9$, see Figure 39. Furthermore, the errors are smaller when $\alpha^n_{g\neq0,r\neq0}$ is used in our adjustment than when $\alpha^n_{g=0,r=0}$ is used.

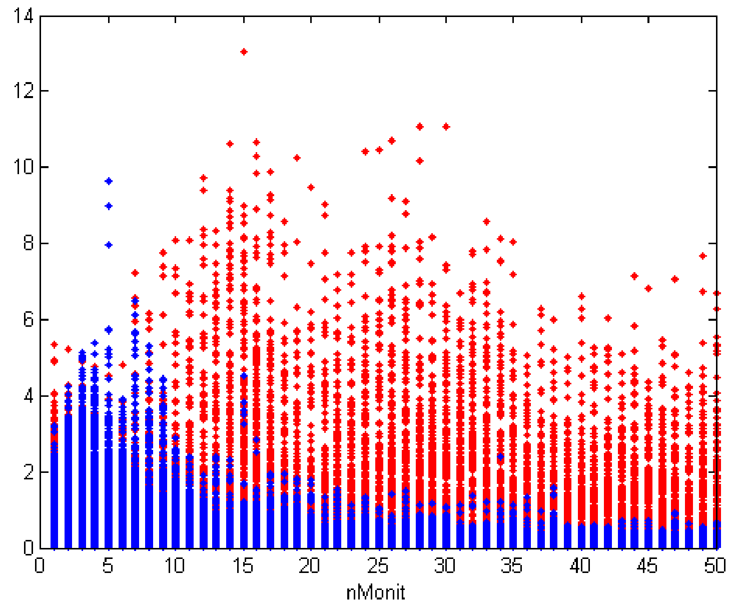

Figure 39: Zoomed in version of Figure 38.

4.3.3.1.4 Analysis with g and r equal to zero (optimal α based on simulated g and r)

In this section we test $\alpha^n_{g\neq0,r\neq0} = 0.618$ on data with $g = 0, r = 0$.

In Figure 40 you can see that using the adjustment still significantly improve the pricing of the barrier option, especially for large original errors. Our adjustment improves the approximation of the price of the barrier option in 871,841 cases out of 880,000. This corresponds to 99.1% ($\alpha^n_{g=0,r=0}$ gave 98.1%).

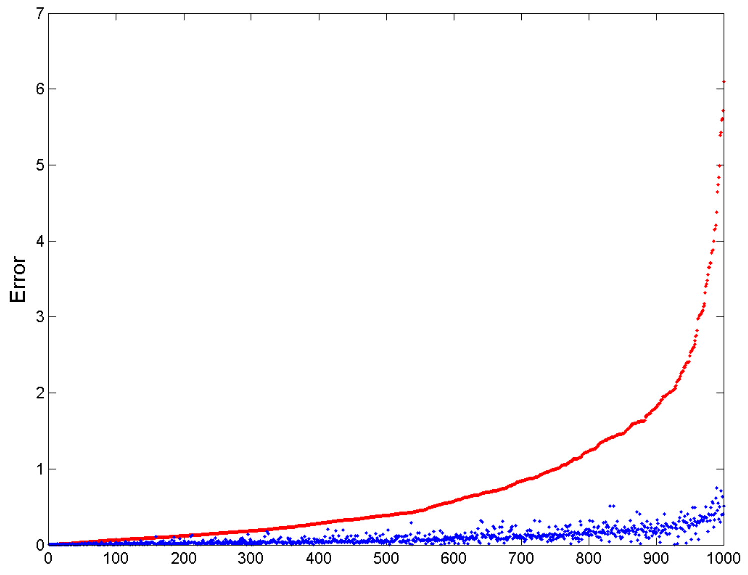

Figure 40: Sorted absolute differences between the true value and the 0.5826-approximation (red) and respective absolute differences between the true value and our adjusted 0.5826-approximation (blue). The plot was obtained using 1,000 simulated parameter sets.

The mean absolute error we get from only using the 0.5826-approximation is 0.737 and the median 0.398. When using the new adjustment we get a mean absolute error of 0.0973 ($\alpha^n_{g=0,r=0}$: 0.0451) and a median of 0.0581 ($\alpha^n_{g=0,r=0}$: 0.0183). The mean improvement is 74.6% ($\alpha^n_{g=0,r=0}$: 75.2%) and the median improvement is 85.8% ($\alpha^n_{g=0,r=0}$: 95.0%).

We can conclude that most performance measures are slightly worse here than in section 4.3.3.1.1, although the percentage of improvements is higher. It is not unexpected that the performance gets worse in general (since we have established in section 4.3.1.3 that the optimal value of α depends on g). The fact that the performance is not substantially worse, however, indicates that the model dependency is not extremely high when it comes to the distribution of g.

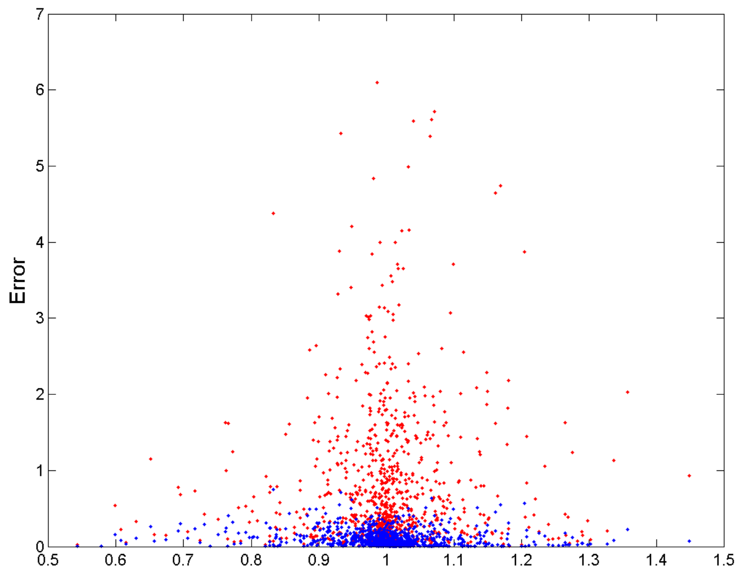

Figure 41: Absolute differences between the true value and the 0.5826-approximation (red) and the absolute differences between the true value and our adjusted 0.5826-approximation (blue) plotted against S/H_A. The plot was obtained using 1,000 simulated parameter sets.

As can be seen in Figure 41, the improvement obtained with the adjustment is especially large when the underlying is close to the adjusted barrier.

There is also in this case almost always an improvement of the result using our adjustment compared to just using the 0.5826-approximation. In Figure 42 you can see the sorted differences between the absolute normalized errors from the plain 0.5826-approximation and the corresponding errors using the 0.5826-approximation with our adjustment (based on $\alpha^n_{g\neq0,r\neq0}$). Once again the "maximum absolute error" (which is used for the normalization) is calculated as the absolute difference between the true value and the 0.5826-approximation, both calculated with the underlying equal to the adjusted barrier. There are still only a few adjustments that give worse results. Specifically, there is an improvement in 99.1% of the cases.

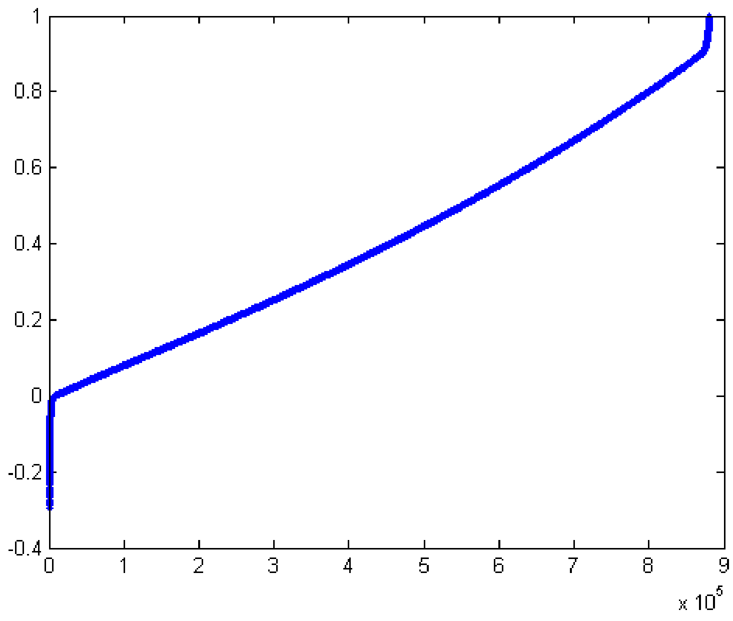

Figure 42: Sorted normalized differences between the absolute differences between the true value and the 0.5826-approximation and the absolute differences between the true value and our adjusted 0.5826-approximation. The plot was obtained using 80,000 simulated parameter sets.

4.3.3.2 Sensitivity analysis of α[34]

By plotting the sorted values of α (see Figure 43) one immediately realizes that, although the majority of the values of α are close to the median, there are large deviations. The aim of this section is to find the reason for this. Are there perhaps certain parameter values which systematically cause abnormal values of α?

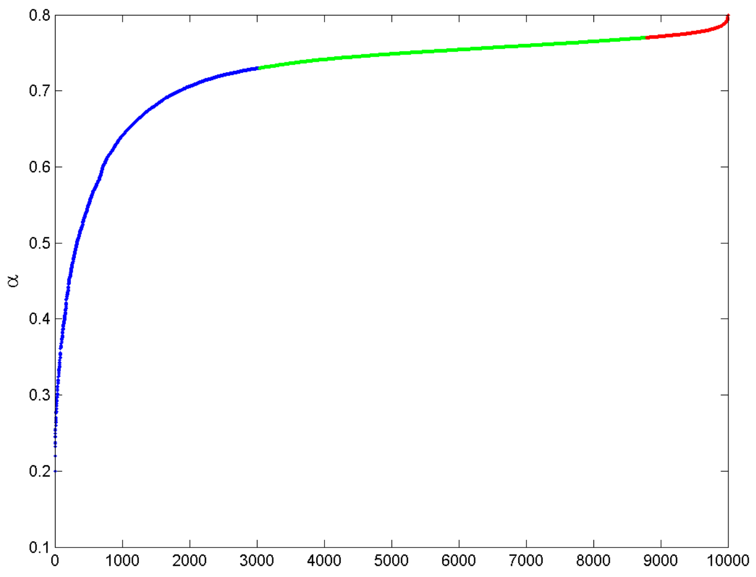

Figure 43: The calculated values of α, sorted in ascending order. Blue represents $\alpha<0.73$, red $\alpha>0.77$, and green the α-values in between.

[34] The analysis in this section has been carried out for 10,000 simulated parameter sets with normally distributed values of the underlying asset.

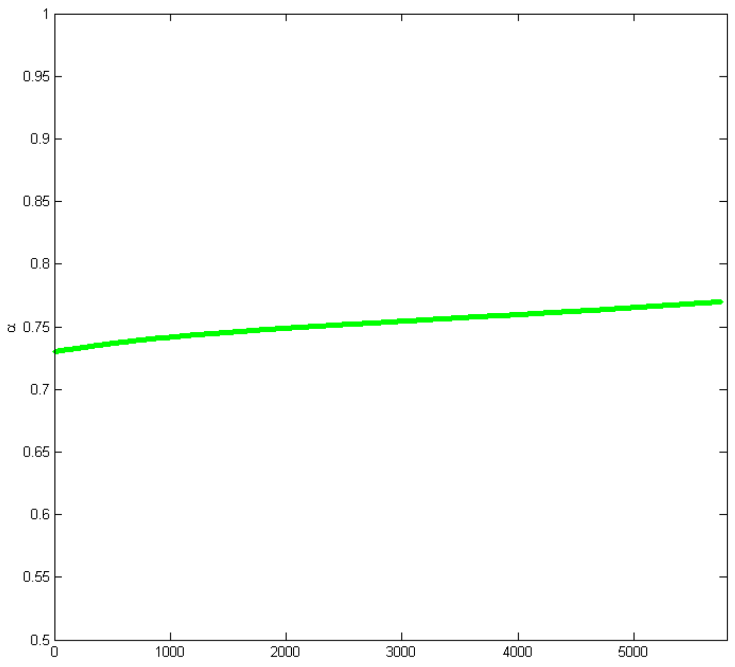

Figure 44: Zoomed in version of Figure 43. The calculated sorted α between 0.73 and 0.77, with a mean of 0.754 and a median of 0.753.

The different groups of α are analyzed by examining correlations between α and the different parameters ($\sigma, T, n_{monit}^{eq}, \Delta\mathrm{t}, H, H_A$).

Table 1: Correlations between α and different parameters.

	σ	T	n_{monit}^{eq}	Δt	H	H_A
α <0.73	0.0564	0.296	0.135	-0.359	0.0273	0.125
0.73<α<0.77	0.00540	-0.0313	-0.0567	-0.222	0.0709	0.128
0.77<α	-0.0277	0.0187	0.0922	-0.0994	0.0177	0.0384

The ideal situation would arguably be that all correlations in Table 1 were equal to zero. That would indicate that all (linear) dependencies are accounted for in the ansatz for the time to maturity of the vanilla call option in our adjustment term.

We can make the observation that the correlations in Table 1 are at least not very large (in absolute terms). It seems that larger values of Δt all else equal give rise to smaller values of α. This is not very strange, because we can expect that the value of α seeks to compensate for large values of Δt (and the other way around), since these values are after all multiplied to obtain the time to maturity of the vanilla call in our adjustment term. We also notice that there is a slight positive correlation between H and α, and a positive correlation also between α and H_A. It is more difficult to draw general conclusions when the correlation changes sign in different intervals, which is the case for all the other parameters.

Based on the correlations in Table 1 and histograms of α with values of other parameters $\left(\sigma, T, n_{monit}^{eq}, \Delta t, H, H_A\right)$ on the horizontal axes, we make hypotheses about what parameter values might cause abnormal values of α (low and high respectively). The dependence of α on S is not analyzed since we calculate each α from 11 different S. We verify these hypotheses using the following two approaches:

1. Extract from the simulated data values of α corresponding to parameter value "profiles" supposedly generating large and small α respectively. Verify that these are on average small/big as expected.
2. Simulate parameter values in accordance with the "profiles" (following very simple distributions, unrelated to the ones used for our main data simulation) and verify that the resulting values of α are on average small/big as expected.

The conclusions presented below rely heavier on the first approach. The plotted values of α have for instance been obtained in that way. The second approach has mainly been used to seek to confirm the conclusions.

Small values of α seem to have a strong representation of $n_{monit}^{eq} = 1$ and $T \in \{0.5, 1\}$, see Figure 45 and Figure 46 below.

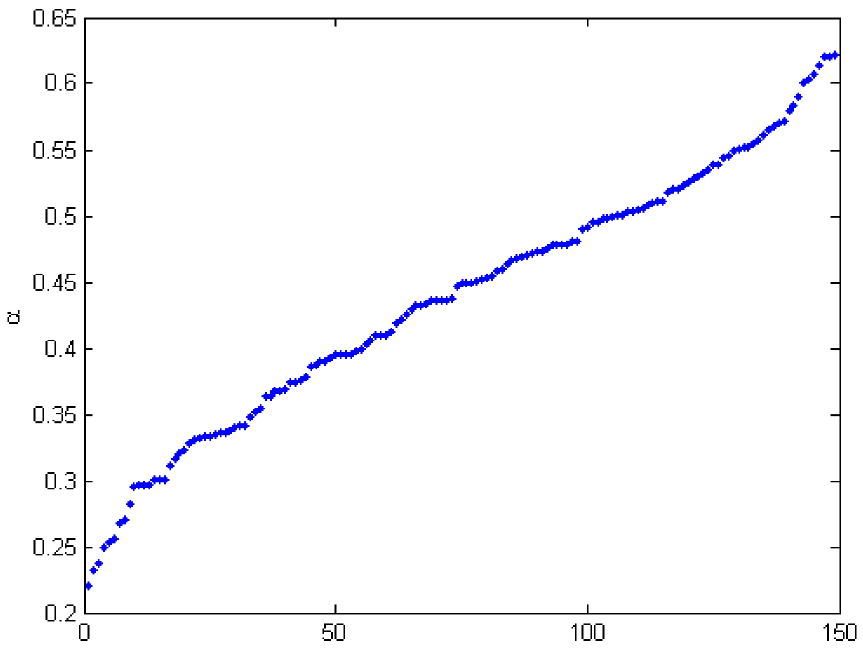

Figure 45: Sorted values of α with n_{monit}^{eq} = 1 and T = 0.5. The mean of these values is 0.438 and the median is 0.449.

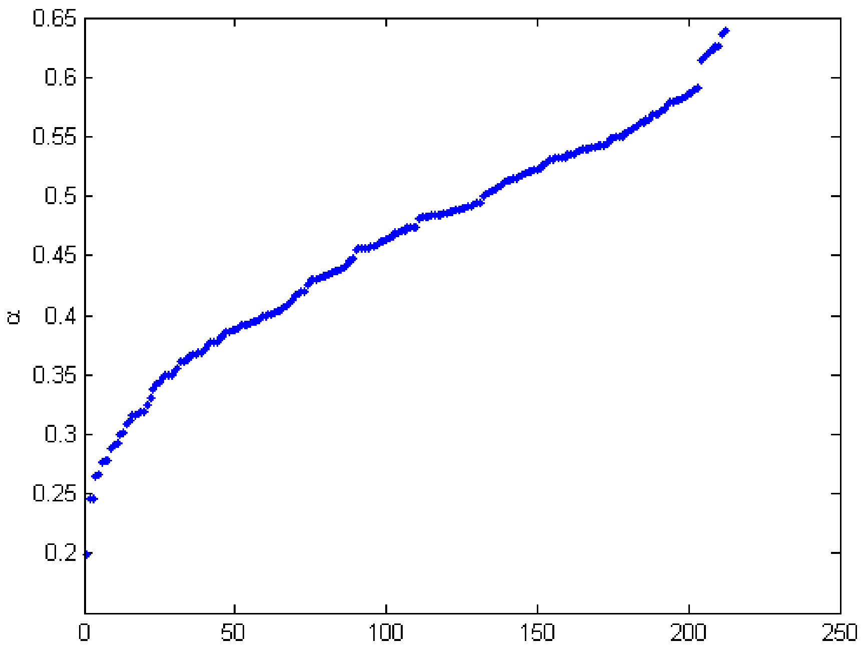

Figure 46: Sorted values of α *with* $n_{monit}^{eq} = 1$ *and* $T = 1$. *The mean of these values is 0.460 and the median is 0.472.*

When it comes to parameter values that systematically tend to give rise to *large* values of α, we have not been able to any tendencies clear enough to present here.

The fact that it was possible to find parameter values that systematically produce abnormal values of α (although only abnormally *low* ones) indicates that α may depend on other parameters in addition to Δt.

4.3.4 Test of potential further improvements of the approximation formula

Based on the findings in the preceding section, a natural question is whether it is possible to improve the adjustment in more ways than simply by finding a better value of α. This is by no means a main part of this thesis, but we nevertheless examine two possibilities in this section. Both are based on linear ansatzes with several parameters included. The first uses a linear ansatz (more extensive than the hypothesized one) for the time to maturity of the vanilla call in the adjustment term, and the second uses a linear ansatz for α. These alternative ansatzes are examined in turn in the following sections. All tests in this section are performed in a similar way as in section 4.3.3.

4.3.4.1 Linear ansatz for time to maturity of vanilla call

In this section we examine whether or not the adjustment is improved if we include more parameters in the ansatz for the time to maturity, T_{call}, of the call option used in the adjustment. We minimize the norm of weighted errors just as described in section 4.3.2.2.1, but with an ansatz of the form

$$T_{call} = a_1\Delta t + a_2\sigma + a_3 T + a_4 H + a_5 r + a_6 g + a_7 S + a_8,$$

and/or

$$\tilde{\Delta}(H_A) = a_9\Delta(H_A).$$

where the only requirement on the coefficients is that they be non-negative and where $\tilde{\Delta}(H_A)$ is used instead of $\Delta(H_A)$ as scale factor in the adjustment term. To test this, we first only allow a_1 to vary in the optimization, let $a_9 = 1$ (fixed), and keep all other coefficients[35] fixed at zero. We then test, in turn and one at a time, having the other coefficients non-fixed (and the rest fixed). Coefficients in the ansatz for T_{call} are, when fixed, always fixed to zero, and a_9 is, when fixed, always fixed to one. We then move on to having *two* coefficients non-fixed at a time, and test all possible combinations there. We continue in this fashion until we reach the case where *all* coefficients are allowed to vary in the optimization. This amounts to 511 tests in total. It should be pointed out that we only include one of the simulated S-values in each parameter set. This is acceptable since we only perform this test for normally distributed values of the underlying[36] (that is, we do not perform the test with symmetrically distributed S). We use (the same) 500 simulated parameter sets for all 511 tests.

It is obviously preferable with as small an objective value[37] as possible. In Figure 47 we see that the magnitudes of these differ between different tests. It is natural to ask oneself whether the different magnitudes have something in common in terms of included coefficients. This is answered in the same figure, and in particular in the zoomed in Figure 48. We see that there is in general an increase in the objective value when the parameter Δt is dropped from the ansatz.

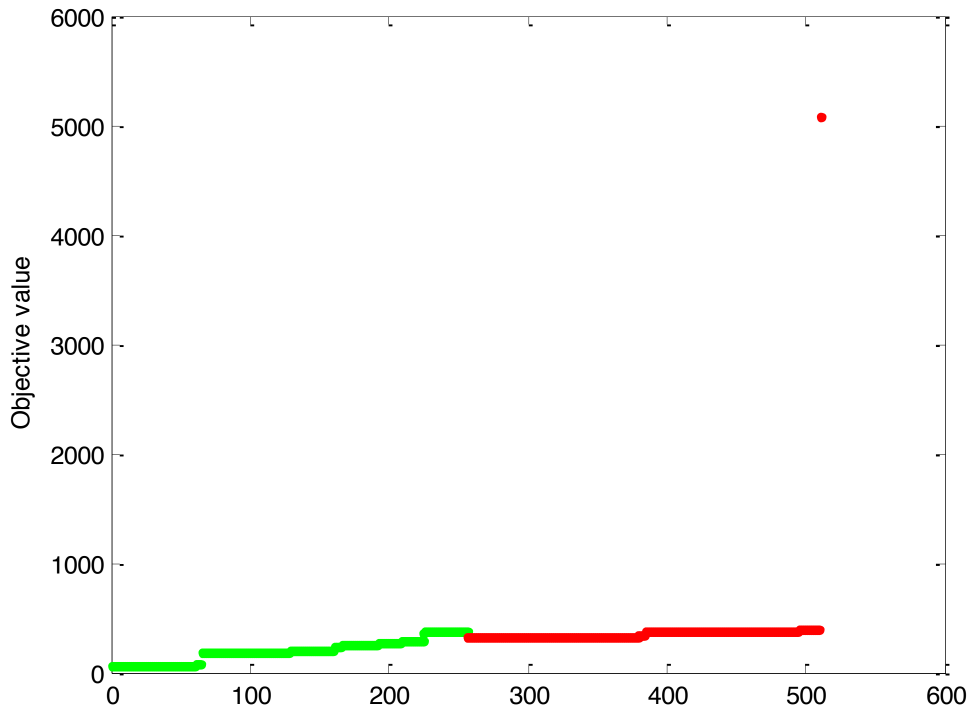

Figure 47: Sorted objective values from the 511 tests. Green dots correspond to tests where a_1 is allowed to vary, and red dots to tests where a_1 is fixed to zero.

[35] In this context, for simplicity, we refer also to a_8 as a coefficient.
[36] The analysis is of course performed on simulated data with both r and g different from zero.
[37] By objective value we mean the norm of weighted errors, since that is what we minimize.

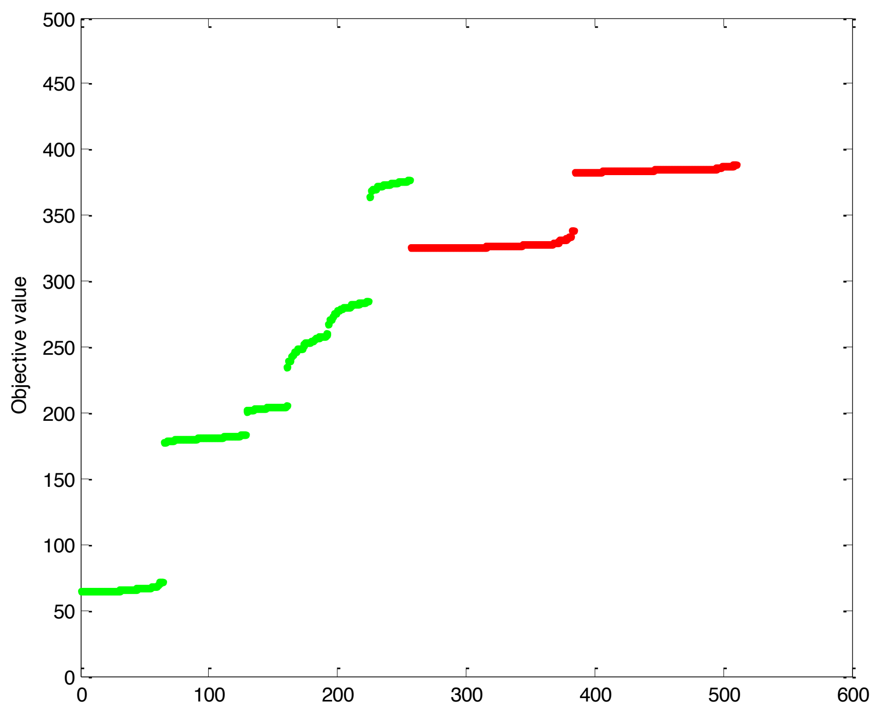

Figure 48: Zoomed in version of Figure 47.

The best value of the objective function, i.e. the smallest value of the 1-norm of the weighted errors, is obtained when

$$\begin{aligned} a_1 &= 0.583, \\ a_2 &= 0.000892, \\ a_5 &= 0.00254, \\ a_9 &= 1.07, \end{aligned} \tag{4.3}$$

and the rest of the coefficients are fixed to zero. The objective value obtained in this case is 64.4. This can be compared to the case when only a_1 is allowed to vary. We then obtain $a_1 = 0.640$ and an objective value of 67.9, 5.4% larger than in the best case.

One might ask oneself how a better objective value can possibly be obtained when we impose some restrictions on the problem than when we do not impose any at all. That is, why is not the case when all coefficients are allowed to vary the one which produces the best objective value?

The answer seems to be surprisingly simple, however unsatisfactory. The fact is that the optimization algorithm (the same variant of *patternsearch* described in section 4.3.2.3) does not reach the global minimum in a high percentage of the cases. The reason is, according to the exit flags the function returns, in all cases the same, namely that it reaches either the maximum number of function evaluations or the maximum number of iterations. This is in spite of the fact that we, as mentioned in section 4.3.2.3, have increased these values substantially above the default ones.

When we take a closer look at the exit flags, it turns out that there are no cases with a_1 included, and at the same time more than four parameters included, where the optimizer has not stopped prematurely because of reaching the maximum number of iterations/function evaluations. On a side note, it can be mentioned that it did stop prematurely even for the “optimal” solution presented above.

We are not quite satisfied with the above result since it is likely that we have not found the global minimum. We therefore multiply the maximum number of function evaluations and iterations by 500 each (meaning that we now have values that are about 10,000 and about 50,000 times the Matlab default values, respectively). Not surprisingly, after running a couple of days, the optimizer finally finds a solution with the best objective value of all we have found, 64.3. The optimal coefficient values are[38]

$$\begin{aligned} a_1 &= 0.583, \\ a_9 &= 1.07, \end{aligned} \tag{4.4}$$

and the rest in the order of 10^{-5} and smaller (although non-zero), and thus considered zero[39]. It can be noticed that these optimal values are very similar to the ones obtained in (4.3), except for the fact that we have considered several coefficient values zero this time. When we test (4.4) in the same way we have done in section 4.3.3 we obtain slight improvements in all measures. We obtain improvements versus the plain 0.5826 approximation in 95.9% of the tested cases (880,000 as usual). The mean improvement is 24.5%, the median improvement 87.6%, the mean absolute error 0.128, and the median absolute error 0.0592.

4.3.4.2 Linear ansatz for α[40]

In this section we examine whether or not the adjustment is improved if we use a linear ansatz for α. We use three approaches – one function that is specified using intuition, one obtained by linear regression, and one obtained by simultaneous optimization of several coefficients (quite similar to section 4.3.4.1).

4.3.4.2.1 Function specified using intuition

We now specify α as a linear function of g of the form

$$\alpha^l = \alpha^n_{g=0,r=0} + (\alpha^n_{g\neq 0,r\neq 0} - \alpha^n_{g=0,r=0})g/\bar{g}, \tag{4.5}$$

where $\bar{g} = 0.0320$ (mean of g in the 20,000 simulated data used to get $\alpha^n_{g\neq 0,r\neq 0}$).

(4.5) is intuitive since α^l is equal to $\alpha^n_{g=0,r=0}$ when $g = 0$ and equal to $\alpha^n_{g\neq 0,r\neq 0}$ when g is equal to its mean value. The linear function is adapted to these two known points. In Figure 49 this linear function is plotted along with all the optimal α's.

[38] Interestingly enough it is noticed that, when rounded to four decimal places, the optimal value of a_1 is actually 0.5826.

[39] One must of course take into account not only the magnitudes of the coefficients, but also the *effects* of these. A small coefficient can have a large effect if the *parameter values* are large. This has been examined as well.

[40] The analysis in this section is done for data with simulated g and r different from zero and with normally distributed values of the underlying.

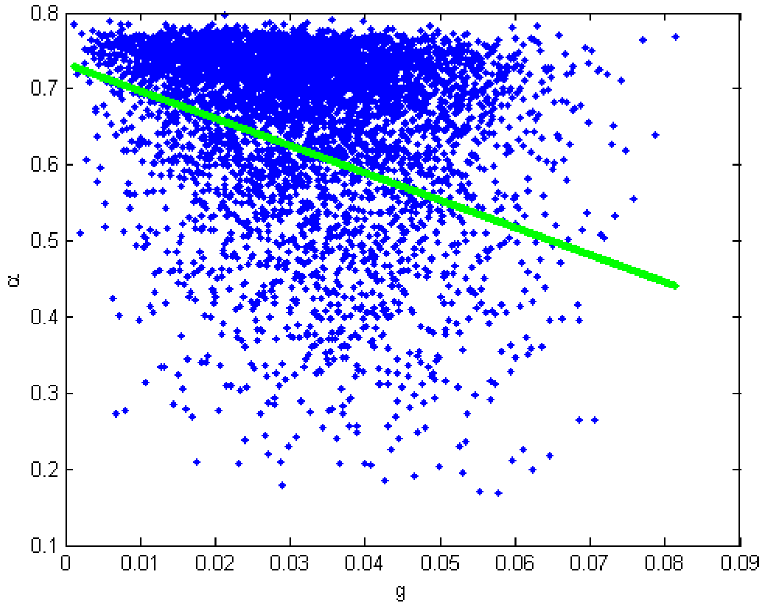

Figure 49: 5,000 optimal α's (one from each simulated parameter set) together with α^l(green) plotted against g.

In Figure 50 you can see that using our approximation in general drastically improves the pricing also in this case. In this test we get an improvement with our adjustment in 844,836 cases out of 880,000 (96.0%).

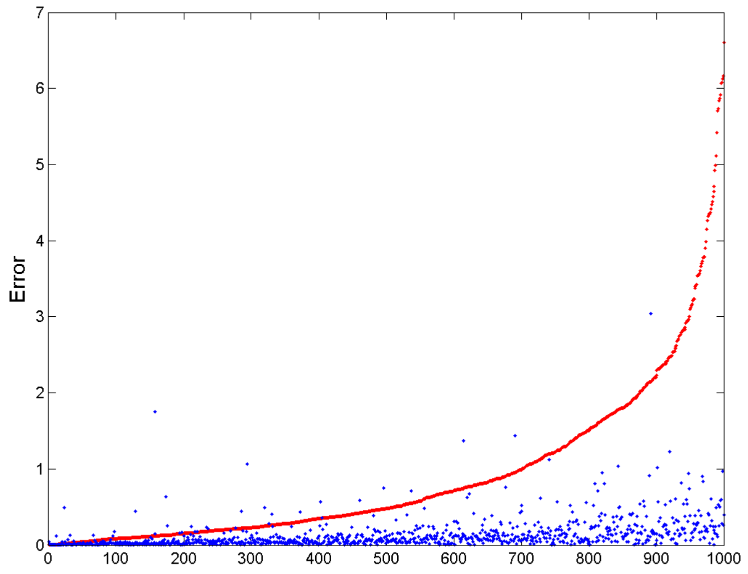

Figure 50: Sorted absolute differences between the true value and the 0.5826-approximation (red) and respective absolute differences between the true value and our adjusted 0.5826-approximation with the adjustment based on α^l(blue). The plot was obtained using 1,000 simulated parameter sets.

When using the adjustment (with α^l) on the 0.5826-approximation we get a mean absolute error of 0.127 (without adjustment: 0.900) and a median of 0.0635 (without adjustment: 0.490). The mean improvement is 23.6% and the median improvement is 86.6%.

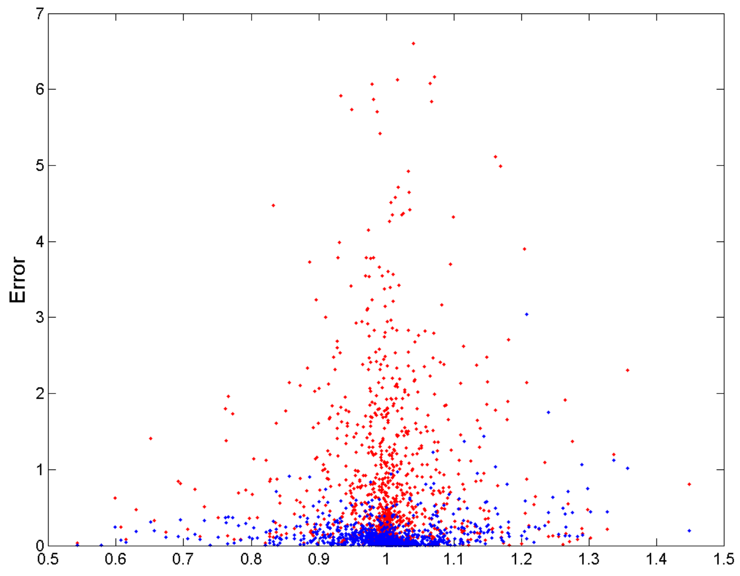

Figure 51: The absolute differences between the true value and the 0.5826-approximation (red) and the absolute differences between the true value and our adjusted 0.5826-approximation with the adjustment based on α^l(blue) plotted against S/H_A. The plot was obtained using 1,000 simulated parameter sets.

Our adjustment still results in substantial improvements when the underlying is close to the adjusted barrier, Figure 51.

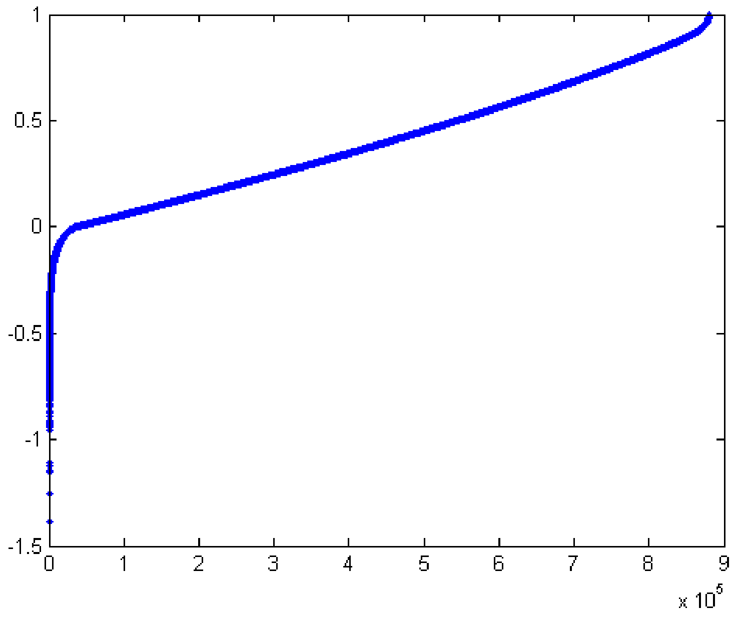

Figure 52: The sorted normalized differences between the absolute differences between the true value and the 0.5826-approximation and the absolute differences between the true value and our adjusted 0.5826-approximation with the adjustment based on α^l. The plot was obtained using 80,000 simulated parameter sets.

Our improvement based on α^l works well for the data with $g \neq 0$ and $r \neq 0$. In Figure 52 you can see the sorted difference between the normalized absolute errors we get by using the plain 0.5826-approximation and the corresponding absolute errors obtained by using the 0.5826-approximation with our adjustment. Once again the "maximum absolute error" (used in the normalization) is calculated as the absolute difference between the true value and the 0.5826-approximation, both calculated with the underlying equal to the adjusted barrier. We get improvements for 96.0% of the simulated parameter sets.

4.3.4.2.2 Function specified using linear regression

Now we try determining the linear function using the Matlab linear regression tool, *polyfit*. We obtain

$$\alpha^{polyfit} = k \cdot g + m,$$

where

$$k = -1.7307,$$
$$m = 0.7172.$$

The fit of this function to the data is seen in Figure 53.

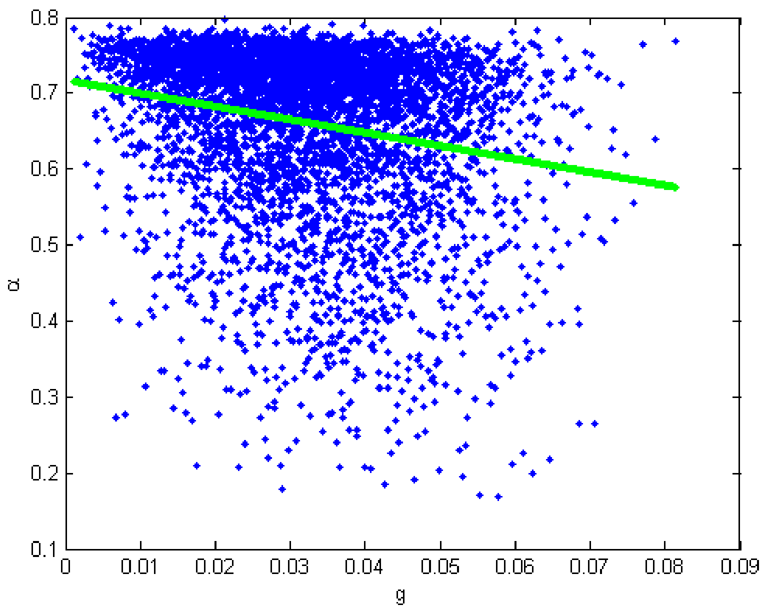

Figure 53: 5,000 optimal α's (one from each simulated parameter set) together with $\alpha^{polyfit}$plotted against g.

When using the adjustment (with $\alpha^{polyfit}$) on the 0.5826-approximation we get an improvement with our adjustment in 836,847 cases out of 880,000 (95.1%). The mean absolute error is 0.123 (without adjustment: 0.900) and the median is 0.0488 (without adjustment: 0.490). The mean improvement is 19.1% and the median improvement is 89.9%.

4.3.4.2.3 Function specified using simultaneous optimization

We want to make a similar test as in section 4.3.4.1, but this time with an ansatz for α instead. We started with 11 coefficients in the ansatz, but unfortunately got memory issues in Matlab when creating all combinations of these. Initial optimizations indicate, however, that the parameters Δt, r, and S may well be dropped from the ansatz. Instead we include H_A, σ^2, and $\frac{g}{\sigma^2}$. The reason that we include the last two "parameters" (terms) is that they are included in the continuous pricing formula. The dependence of α (or lack thereof) on the two last ones are to some extent illustrated in Figure 54 and Figure 55. Although the dependence does not seem particularly strong, we decide to proceed with these "parameters"[41] in the ansatz to be tested. The ansatz to be tested is therefore

$$\alpha = b_1\sigma + b_2T + b_3H + b_4g + b_5H_A + b_6\sigma^2 + b_7\frac{g}{\sigma^2} + b_8.$$

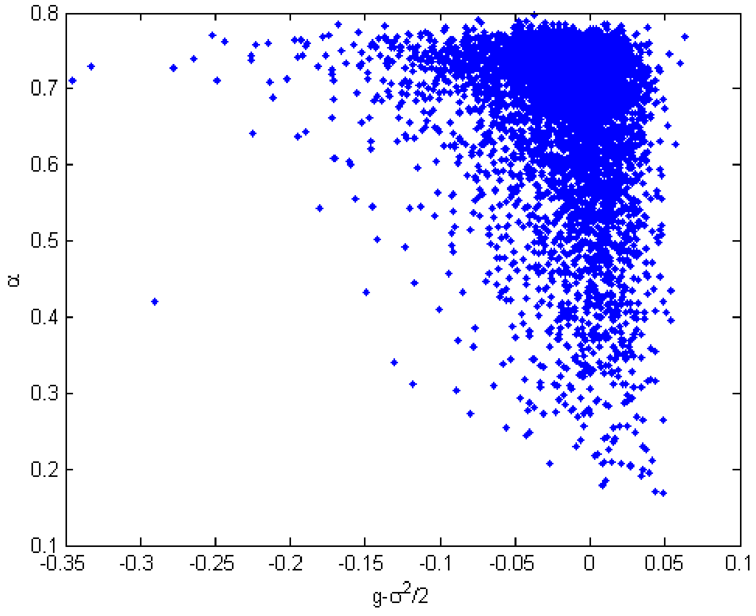

Figure 54: 5,000 optimal α's plotted against $(g - \frac{\sigma^2}{2})$.

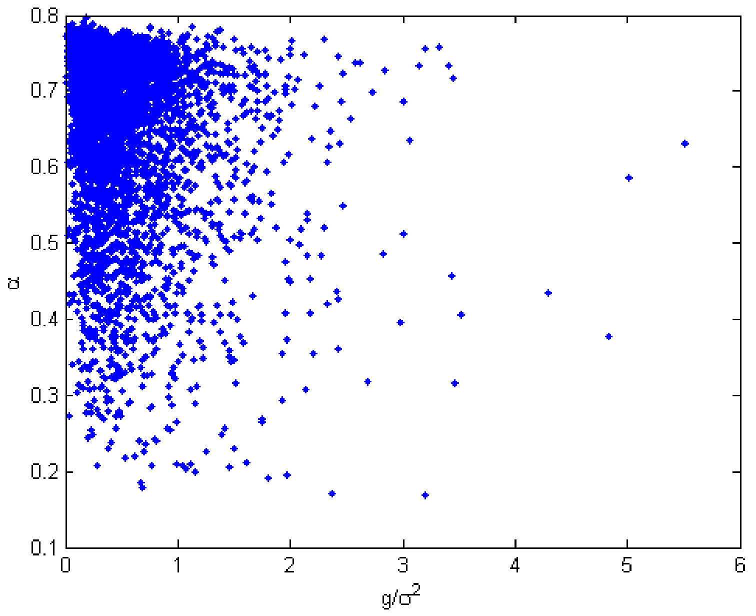

Figure 55: 5,000 optimal α's plotted against $\frac{g}{\sigma^2}$.

Since we established in section 4.3.4.1 that the best solution seems to be obtained when all parameters are included (provided that we increase the maximum number of function evaluations and iterations

[41] We see no reason to explicitly include $g - \frac{\sigma^2}{2}$, however. We consider it better to let the optimizer find this combination itself should it be a good one.

enough, and we do it as much as in that section) we decide to only optimize this case here. The optimal coefficient values obtained are

$$\begin{aligned} b_1 &= 0.418, \\ b_2 &= 0.0129, \\ b_5 &= 0.00577, \\ b_8 &= 0.00145, \end{aligned}$$

and the rest practically (although not exactly) equal to zero[42]. We obtain improvements versus the plain 0.5826 approximation in 96.3% of the tested cases (880,000 as usual). The mean improvement is 26.4%, the median improvement 87.1%, the mean absolute error 0.118, and the median absolute error 0.582.

4.4 Conclusion for the first problem

We began the analysis of the first problem by making some initial tests. It was validated that there is in fact a relationship that makes the ansatz for $\hat{T}$ reasonable, and that the optimal value of α is independent of the risk free rate. We realized, on the other hand, that the cost of carry cannot be excluded from the analysis.

We formulated two different optimization problems, determined (with an approach based on Matlab documentation studies in combination with trial and error) suitable optimization algorithms, made analyses to decide on an adequate number of simulations, and finally determined optimal values of the coefficient α in the ansatz $\hat{T} = \alpha\Delta t$, where $\hat{T}$ denotes the time to maturity of the vanilla call option in the adjustment proposed in the hypothesis.

We tested the hypothesis and specifically the obtained optimal values of α (over 3.5 million tests in total, divided on four cases). We notably obtained improvements versus the plain 0.5826-approximation in well over 90 percent of the tests in all four cases, and median improvements of 95, 91, 87, and 86 percent in the four tested cases respectively.

Our overall conclusion is that the proposed adjustment works very well with our choices of the coefficient α. Below we provide the complete suggested approximation formula when $\Delta t^{first} = \Delta t$.

$$V^{eq} = C_{DO}^{0.5826}(T, S, X, H, \sigma, r, g, \Delta t) + adjustment,$$

with

$$adjustment = \Delta(H_A) \cdot \left(C\left(\hat{T}, S, \hat{X}, \sigma, \hat{r}, g\right) - max\left(S - \hat{X}, 0\right)\right),$$

where

$$\begin{gathered} \hat{r} = 0, \\ \hat{T} = \alpha\Delta t, \\ \hat{X} = H_A = He^{-0.5826\sigma\sqrt{\Delta t}}, \\ \Delta(H_A) = \frac{\partial C_{DO}^{0.5826}}{\partial S}(H_A), \\ \alpha = \begin{cases} \alpha_{g=0,r=0}^{n} = 0.733, & if\ g = r = 0, \\ \alpha_{g\neq 0,r\neq 0}^{n} = 0.618, & if\ g \neq 0,\ r \neq 0. \end{cases} \end{gathered}$$

In the case with g different from zero more model dependency is introduced. Since the value of α is different for different values of g (as shown in section 4.3.1.3), the optimal value obtained in our tests will of course depend on the distribution of our simulated g-values. In our opinion we have made quite

[42] They are in the order of 10^{-13}, and probably a result of machine precision.

reasonable assumptions regarding the distribution, but it is nevertheless a source of error[43]. Although a cost of carry equal to zero is not realistic, because of less model dependency we consider the formula obtained in this case our main result.

One way to decrease the model dependency for the case of non-zero cost of carry may be to use one of the potential further improvements examined in section 4.3.4. The best overall results were obtained for the ansatz for α in section 4.3.4.2.3, but it might be that the "intuitive" formula tested in section 4.3.4.2.1 performs better in a "real-world" situation. This might be a subject of further research.

[43] It should in particular be noted that we have not examined cases with negative cost of carry.

5 Problem 2: Approximation inaccurate when $\Delta t^{first} \neq \Delta t$

The aim of this section is to improve the 0.5826-approximation, in particular close to the (adjusted) barrier. We no longer assume that $\Delta t^{first} = \Delta t$, but rather that $\Delta t^{first} < \Delta t$.[44]

5.1 Problem description

In the 0.5826-approximation it is assumed that the time steps between the monitoring instants are equidistant. This is in practice seldom the case. A common situation is when the *first* time step is of a different size (in particular *smaller*) whereas the others are equidistant. This will for instance be true for any discretely monitored barrier option during most of its "life" since the time to the first monitoring instant changes continuously as time passes (until that monitoring instant is reached and the next monitoring instant becomes the new "first"). This will obviously, in a more or less significant way, change the (true) value of the barrier option. An example can be seen in Figure 56.

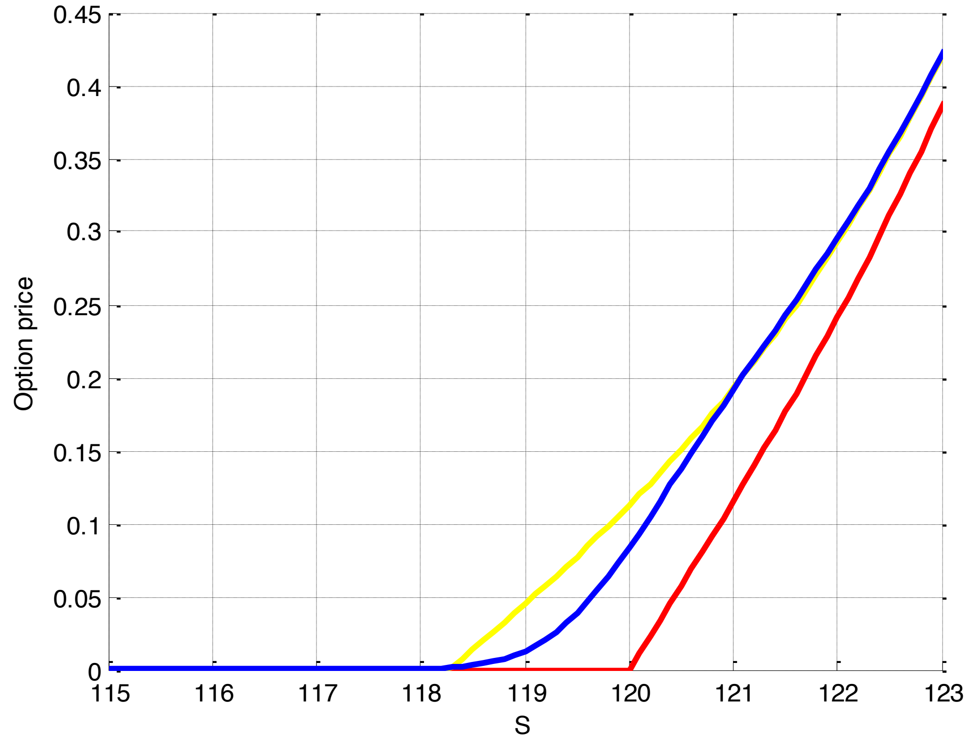

Figure 56: The blue curve depicts the true price of a discretely monitored down-and-out call option with $\Delta t^{first} \neq \Delta t$, the yellow curve the 0.5826-approximated price, and the red curve the price of the corresponding continuously monitored option, for different prices of the underlying asset, S. The following parameter values have been used: $X = 130, g = r = 0, \sigma = 0.1, T = 0.25, H = 120, \Delta t \approx 0.0615, \Delta t^{first} = 0.25/65$. This corresponds to $H_A \approx 118.3$.

How much the value changes does of course depend on how much the first time step differs in size and how close to the barrier the price of the underlying is. It is for instance easily understood that the price of the down-and-out call will be very close to zero when the price of the underlying is close to, and below, the barrier, given that $\Delta t^{first} \ll \Delta t$ (and very close to zero). This is not reflected by the 0.5826-approximation formula.

[44] It is easily understood that equality would turn problem two into problem one.

5.2 Hypothesis

The hypothesized solution to this problem is to calculate two adjusted values, both calculated using the method proposed as solution to problem one. One of the values, V^{eq}, corresponds to all the later monitoring instants (with equidistant time steps between them). It uses Δt as the time between monitoring instants "as usual" in the adjusted approximation formula developed in the preceding sections. The other value, V^{one}, corresponds to the first monitoring instant, and uses Δt^{first} (the time to the first monitoring instant) as the time between monitoring instants in the adjusted approximation formula. The computed option value according to this hypothesis is a weighted sum of these two values, where the weight factor is chosen as the value of an appropriate digital call option (defined below)[45]. Formally written, this hypothesis becomes

$$V = kV^{eq} + (1-k)V^{one},$$

with

$$k = C_{digital}(\hat{T}_{digital}, S, \hat{X}_{digital}, \sigma, \hat{r}, g)$$

where

$$\hat{T}_{digital} = \gamma \Delta t^{first},$$
$$\hat{X}_{digital} = H,$$
$$\hat{r} = 0.$$

A schematic example of the hypothesized approximation at work can be seen in Figure 57. The price of a digital call option is shown in Figure 58.

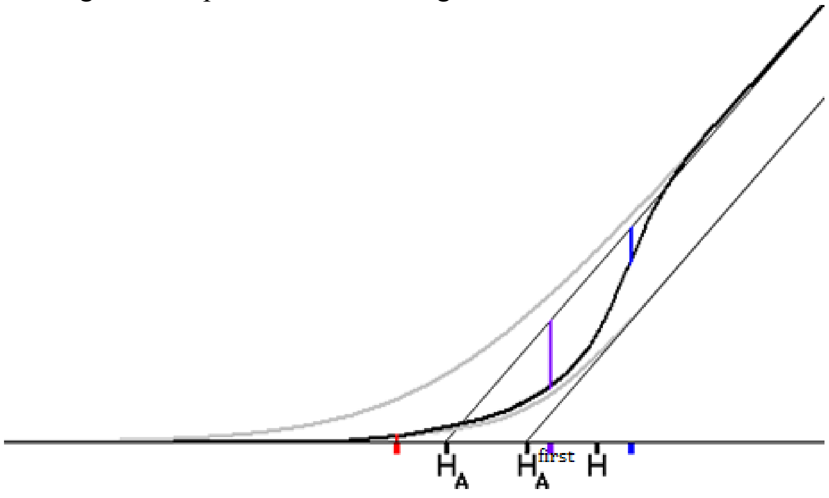

Figure 57: Weighted adjustment when the stub to the first monitoring instant is about 25% of the time step between the other monitoring instants.

[45] The type of digital option used is a cash-or-nothing call that pays out either one currency unit or nothing at maturity (depending of course on whether or not it is in the money).

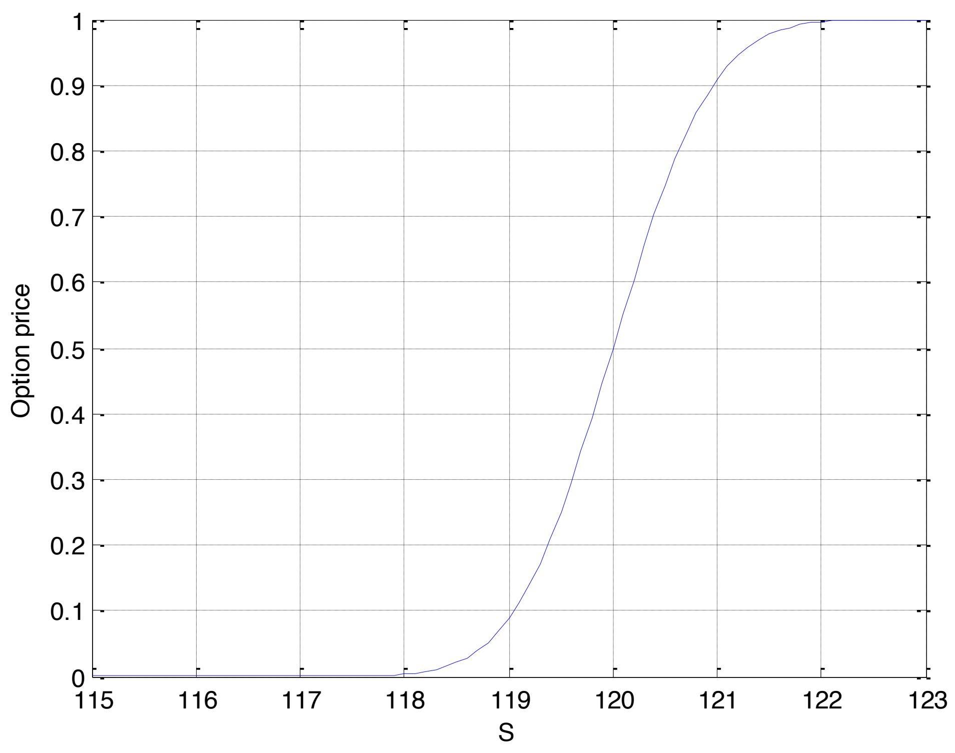

Figure 58: The price of a digital option, plotted for different prices of the underlying asset, S*. The following parameter values have been used:* $g = 0, \sigma = 0.1, T = 0.25, \hat{X}_{digital} = 120$.

5.2.1 Rationale for the hypothesis

The rationale for this hypothesis is that the contribution from the term corresponding to the equidistant time steps should be close to zero when it is very likely that the barrier will be hit at the first monitoring instant, and the other way around when the opposite is true. The value of the digital option will for instance be very close to one if the time to the first monitoring instant is very short and it is in-the-money or if that time is somewhat longer and it is fairly deep in-the-money. It is therefore intuitive to test it as weight factor (compare Figure 58).

The idea is thus more or less to take an expectation of the option value, given the probabilities that the price of the underlying will be above or below the barrier at the first monitoring instant. The risk-neutral probability of a price of the underlying asset above the barrier at the first monitoring instant, given an initial value $S_0 = s$, is given by

$$Q\left(S_{\Delta t^{first}} > H | S_0 = s\right) = Q\left(se^{\left(g-\frac{\sigma^2}{2}\right)\Delta t^{first}+\sigma W_{\Delta t^{first}}} > H\right) = Q\left(\left(g - \frac{\sigma^2}{2}\right)\Delta t^{first} + \sigma\sqrt{\Delta t^{first}}G > \log\left(\frac{H}{s}\right)\right) = Q\left(G > \frac{\log\left(\frac{H}{s}\right)-\left(g-\frac{\sigma^2}{2}\right)\Delta t^{first}}{\sigma\sqrt{\Delta t^{first}}}\right) = \Phi(d),$$

with

$$d = \frac{\log\left(\frac{s}{H}\right)+\left(g-\frac{\sigma^2}{2}\right)\Delta t^{first}}{\sigma\sqrt{\Delta t^{first}}},$$

and where Q denotes the risk neutral probability measure, Φ denotes the standard normal cumulative distribution function and $G \sim N(0,1)$.[46]

It is immediately noticed that the above probability is equal to the price of a digital call (with the risk free rate set to zero) with time to maturity Δt^{first}, strike H, and the value of the other parameters as expected.[47]

This is our rationale for the hypothesis, and then the coefficient γ for Δt^{first} is included in the ansatz for $\hat{T}_{digital}$ to examine whether optimization with respect to this coefficient can improve the hypothesized approach further.

5.3 Analysis and results

In this section we test, and manage to improve, the adjustment outlined in the previous section and find an optimal value of the parameter γ in the ansatz for $\hat{T}_{digital}$.

We make three main iterations in the analysis of this problem. That is because we find that the first approach (although already slightly changed compared to the initial hypothesis) does not produce altogether satisfactory results. We will consequently divide this section in three, one for each iteration. Each of these comprises the following sections:

1. Determination of optimal γ[48]
2. Test of optimal γ

The optimal values of γ obtained in the following optimizations have superscript *n* (norm), *r* (ratio), *p* (percentage), and *m* (mean), respectively.

5.3.1 First iteration

Our first approach is to stick with the general methodology for the *first* optimization approach developed for problem one (we do not to proceed with the second one).

5.3.1.1 Determination of optimal γ

The analysis in this section follows these steps:

1. State optimization problems.
2. Determine optimal number of simulated values.
3. Determine optimal values of γ.

The steps are described in turn below.

5.3.1.1.1 Optimization problem

The idea in this approach is to minimize the 1-norm of a vector of weighted errors between the true value and the approximated value using our proposed approximation, i.e. to obtain *one* optimal value of γ for n simulated parameter sets (the same approach used in section 4.3.2.2.1 for problem one). We solve the following problem numerically:

$$\min_{\gamma}\left[\left\|\boldsymbol{C}_{DO}^{FD} - \boldsymbol{V}\right\|_1\right] over\ all\ \gamma \geq 0. \tag{5.1}$$

[46] We assume that the reader is familiar with stochastic calculus, and in particular knows how to solve (1.1). A possible reference is otherwise [9, p. 69].
[47] The reader who wants this confirmed is referred to [8].
[48] We will not describe how option values and deltas were obtained from the simulated data, because that was done in more or less exactly the same way as for problem one. We therefore simply refer to section 4.3.2.1. Similarly, for discussions of the choice of optimization algorithm we refer to section 4.3.2.3, since we use different versions of the Matlab function *patternsearch* also for the second problem.

The problem can be rewritten more explicitly (and in scalar form) as

$$\min_{\gamma}\left[\sum_{i=1}^{n}\sum_{j=1}^{m}\left|C_{DO}^{FD}\left(T_i,S_{i,j},X,H_i,\sigma_i,r_i,g_i,n_{monit,i}^{eq},n_{steps}\right)-approximation_{i,j}\right|\right] over\ all\ \gamma \geq 0,$$

where

$$approximation_{i,j} = k_{i,j}V_{i,j}^{eq} + \left(1-k_{i,j}\right)V_{i,j}^{first},$$

$$V_{i,j}^{eq} = C_{DO}^{0.5826}\left(T_i,S_{i,j},X,H_i,\sigma_i,r_i,g_i,n_{monit,i}^{eq}\right)+\Delta_i\left(H_{A,i}\right) \cdot \left(C(\hat{T}_i,S_{i,j},\hat{X}_i,\sigma_i,r_i,g_i)-max(S_{i,j}-\hat{X}_i,0)\right),$$

$$V_{i,j}^{first} = C_{DO}^{0.5826}\left(T_i,S_{i,j},X,H_i,\sigma_i,r_i,g_i,n_{monit,i}^{first}\right)+\Delta_i\left(H_{A,i}\right) \cdot \left(C(\hat{T}_i^{first},S_{i,j},\hat{X}_i^{first},\sigma_i,r_i,g_i)-max(S_{i,j}-\hat{X}_i^{first},0)\right),$$

$$k_{i,j} = C_{digital,i,j}\left(\hat{T}_{digital,i},S_{i,j},X_i,\sigma_i,r_i,g_i\right),$$

with

$$n_{monit,i}^{eq} = \frac{T_i}{\Delta t_i},$$

$$n_{monit,i}^{first} = \frac{T_i}{\Delta t_i^{first}},$$

$$\hat{T}_i^{first} = \alpha\Delta t_i^{first},$$

$$\hat{X}_i^{first} = H_{A,i}^{first},$$

$$H_{A,i}^{first} = H_i \cdot e^{-0.5826\sigma\sqrt{\Delta t_i^{first}}},$$

$$\hat{T}_{digital,i} = \gamma\Delta t_i^{first}.$$

The attentive reader may notice two unexpected things; the absence of the weight factor, ***l***, from problem one and a slight change compared to the presented hypothesis – both deltas are now the same. We briefly cover these issues in turn.

We have chosen to leave out the weight factor, ***l***, which was used for the first problem, from (5.1). This is because we simulate the values of the underlying in a quite narrow interval around the barrier, and in addition the values of the underlying are normally distributed around H. We therefore see no reason to further diminish the importance of the errors the farthest from H (which would have been the effect had the weight factor been included).

We have replaced V^{one} with V^{first}. The only difference between the two is that V^{first}, just as V^{eq}, has a scaling factor equal to $\Delta(H_A)$. The main reason for this is that we obtain better objective values in the optimization in this way, both when $g = 0, r = 0$ (7.21% improvement) and when they are non-zero (1.99% improvement). Most of the tests performed (mean absolute error etc.) also indicate that this new approach is preferable. An additional benefit is of course that efficiency is slightly improved since we save a lot of delta calculations.[49]

[49] The *very* first approach was to test the initial hypothesis. We do not present those results in the thesis, however.

5.3.1.1.2 Optimal number of simulated parameter sets

A similar analysis as for problem one has been carried out to find the optimal number of simulated parameter sets. Our conclusion from Figure 59 is that there is no reason not to stick with 20,000 simulations also for this second problem. The γ determined from 20,000 simulations is only 0.4% smaller than the one determined based on 100,000 simulations.

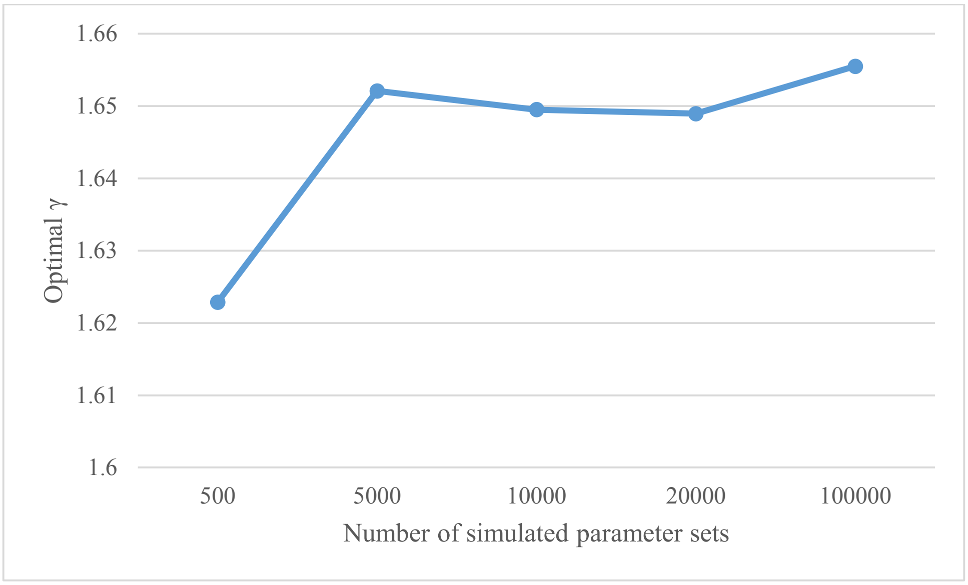

Figure 59: Optimal values of γ, for different numbers of simulated parameter sets.

5.3.1.1.3 Optimal γ

The optimal values of γ obtained in this iteration are

$$\gamma^n_{\mathrm{g=0,r=0}} = 1.65,$$

$$\gamma^n_{\mathrm{g\neq0,r\neq0}} = 3.57.$$

5.3.1.2 Test of optimal γ

In this section we test the adjusted approximation based on the optimal values of γ on data that have not been used to obtain these optimal values. 80,000 simulated parameter sets are used in the test data in all tests (although of course only about half of these are below the barrier). $\gamma^n_{\mathrm{g=0,r=0}}$ is tested on data with $r = 0$, $g = 0$ using $\alpha^n_{g=0,r=0}$, and $\gamma^n_{\mathrm{g\neq0,r\neq0}}$ is tested on data with $r \neq 0$ and $g \neq 0$ using $\alpha^n_{g\neq0,r\neq0}$. The improvement is analyzed and we show how well the adjustment works for different weights.

5.3.1.2.1 Analysis of $\gamma^n_{\mathrm{g=0,r=0}}$

The optimal value of γ obtained in the maximization of the percentage of improvements is $\gamma^n_{\mathrm{g=0,r=0}} = 1.65$. This value is tested in the current section.

The adjustment improves the pricing of the barrier options in 615,996 cases out of 880,000 (70%). This can be seen in Figure 60 where some "new" errors are larger than the "old" ones.

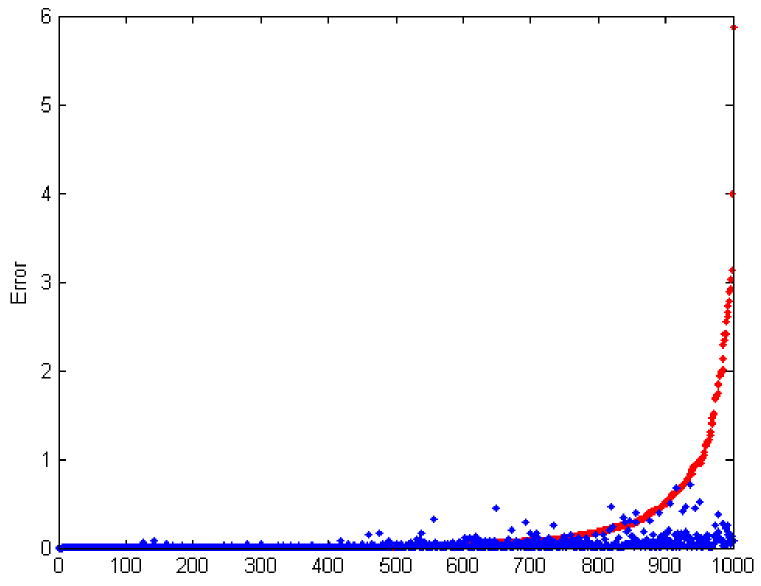

Figure 60: Sorted absolute differences between the true value and the 0.5826-approximation (red) and respective absolute differences between the true value and our adjusted 0.5826-approximation (blue). The plot was obtained using 1,000 simulated parameter sets.

When using the adjustment on the 0.5826-approximation we get a mean absolute error of 0.0323 (without adjustment: 0.175) and a median of 0.0077 (without adjustment: 0.0142). The median improvement is 33.3% when using the weighted adjustment together with the 0.5826-approximation compared to just using the 0.5826-approximation.

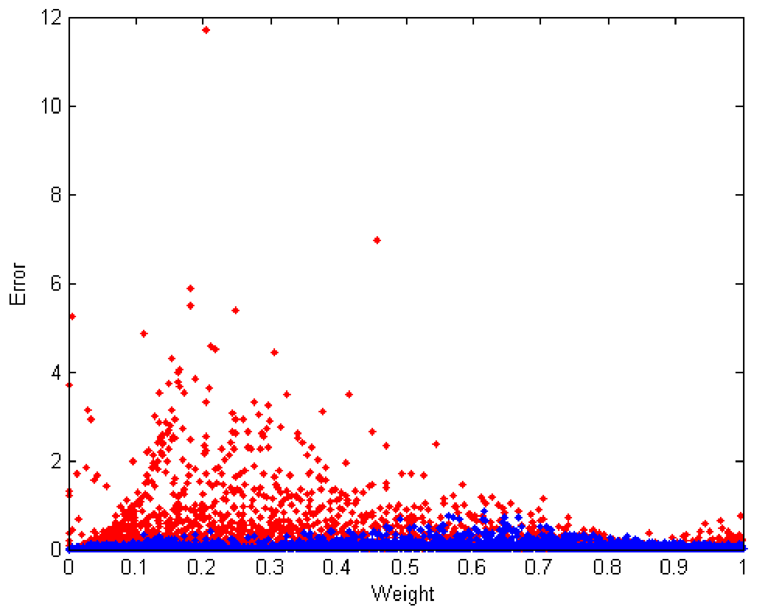

Figure 61: Absolute differences between the true value and the 0.5826-approximation (red) and the absolute differences between the true value and our adjusted 0.5826-approximation (blue) plotted against the weight. The plot was obtained using 5,000 simulated parameter sets.

When comparing absolute errors with and without our adjustment, it can be noted that a significant improvement is in general achieved for small weights. It is on the other hand more common with small or no improvements, or even worse errors, for larger weights (see Figure 61).

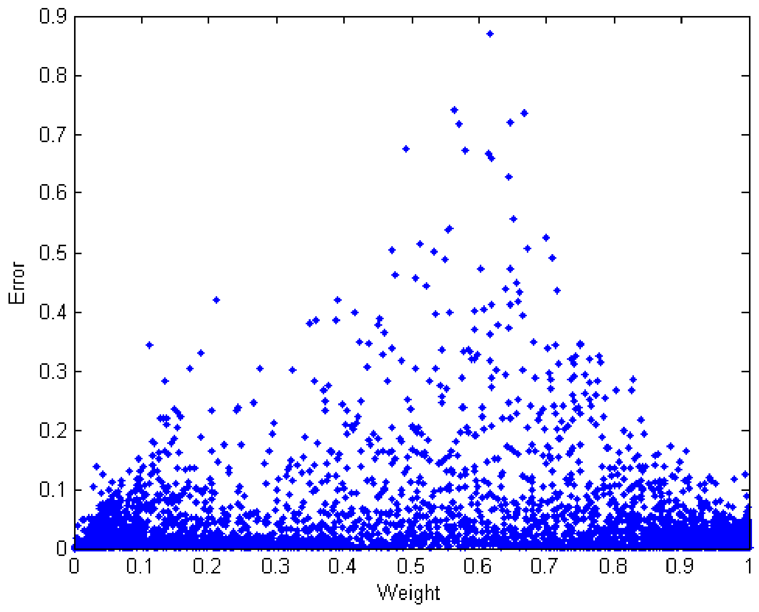

Figure 62: Absolute differences between the true value and our adjusted 0.5826-approximation (blue) plotted against the weight. The plot was obtained using 5,000 simulated parameter sets. This is in fact Figure 61 a bit zoomed in and without the errors corresponding to the plain 0.5826-approximation.

We did suspect that our new weighted adjustment would give the largest absolute errors when the weight is close to 0.5. Our reason for this was that the underlying is then close to the barrier, meaning that the value of the digital option (and equivalently the weight) changes a lot even for small movements in the price of the underlying. This suspicion is confirmed in Figure 62 where the largest absolute errors are in the middle, as expected. Most weights are close to zero and one, meaning that the adjustment often uses either the adjustment based on Δt^{first} or the one based on Δt almost exclusively.

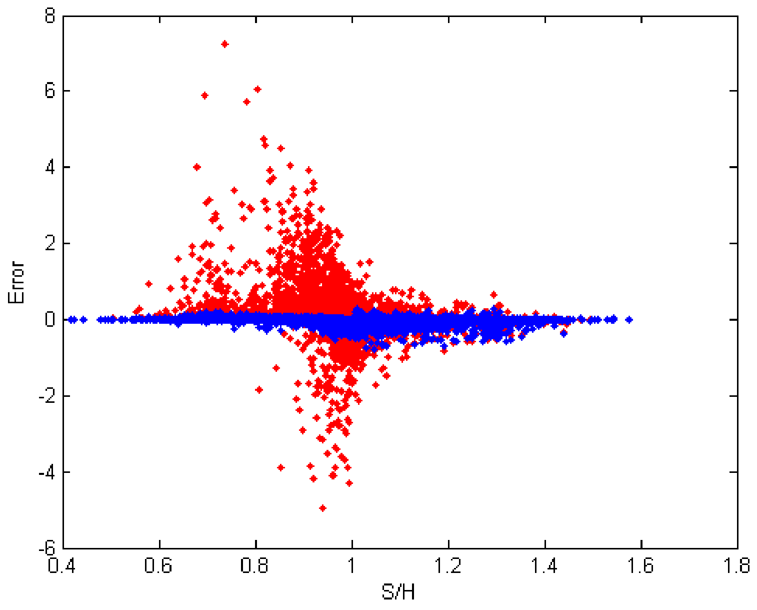

Figure 63: The difference between the true value and the price using our adjusted 0.5826-approximation (blue) and the difference between the true value and the 0.5826-approximation (red) for discretely monitored down-and-out call options, plotted against $\frac{S}{H}$. The plot was obtained using 1,000 simulated parameter sets.

The new adjustment improves the approximation of the price significantly when $S/H < 1$ in the case with $g = 0, r = 0$, as can be seen in Figure 63. Above the limit the adjustment does not seem to improve the approximation of the price as much.

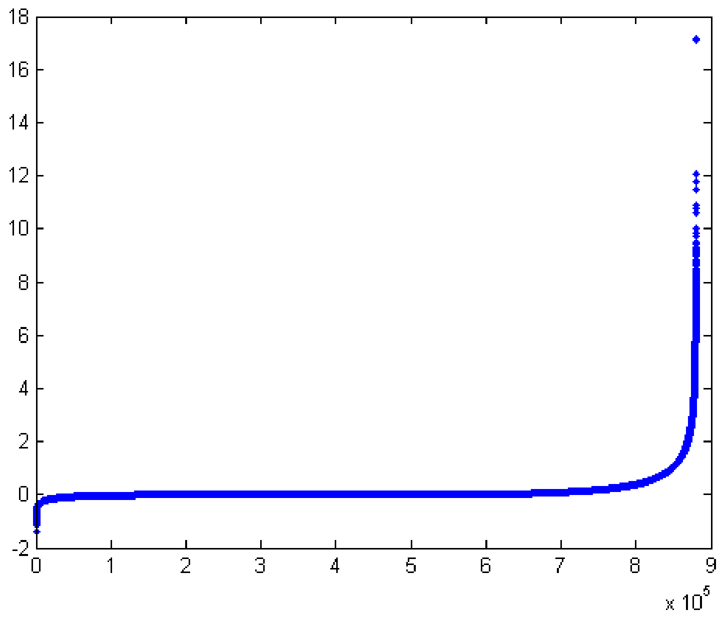

Figure 64: Sorted differences between the absolute differences between the true value and the 0.5826-approximation and the absolute differences between the true value and our adjusted 0.5826-approximation. The plot was obtained using 80,000 simulated parameter sets.

It is seen in Figure 64 that we get a great result but in some cases our new adjustment gives larger absolute errors.

5.3.1.2.2 Analysis of $\gamma^{n}_{\mathrm{g}\neq 0,\mathrm{r}\neq 0}$

The optimal value of γ obtained in the maximization of the percentage of improvements is $\gamma^{n}_{\mathrm{g}\neq 0,\mathrm{r}\neq 0} =$ 3.57. This value is tested in the current section.

As in the $g = 0, r = 0$ case the approximation of the barrier option price is improved using our adjustment. We get an improvement with our weighted adjustment in 501,500 cases out of 880,000 (57.0%). In Figure 65 you can see that the “new” errors are larger than the “old” ones in more cases than in the $g = 0, r = 0$ case.

When we use the weighted adjustment on the 0.5826-approximation we get a mean absolute error of 0.171 (without adjustment: 0.214) and a median absolute error of 0.0284 (without adjustment: 0.0171). The median improvement is 6.22%.

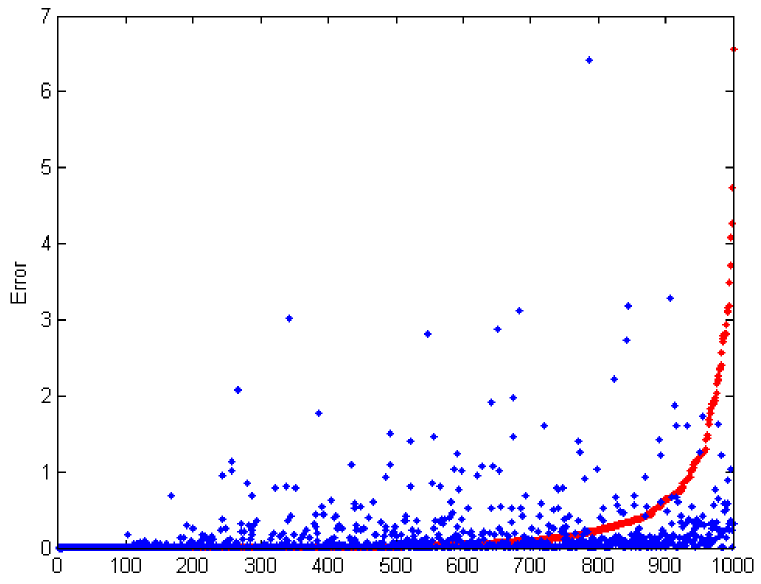

Figure 65: Sorted absolute differences between the true value and the 0.5826-approximation (red) and respective differences between the true value and our adjusted 0.5826-approximation (blue). The plot was obtained using 1,000 simulated parameter sets.

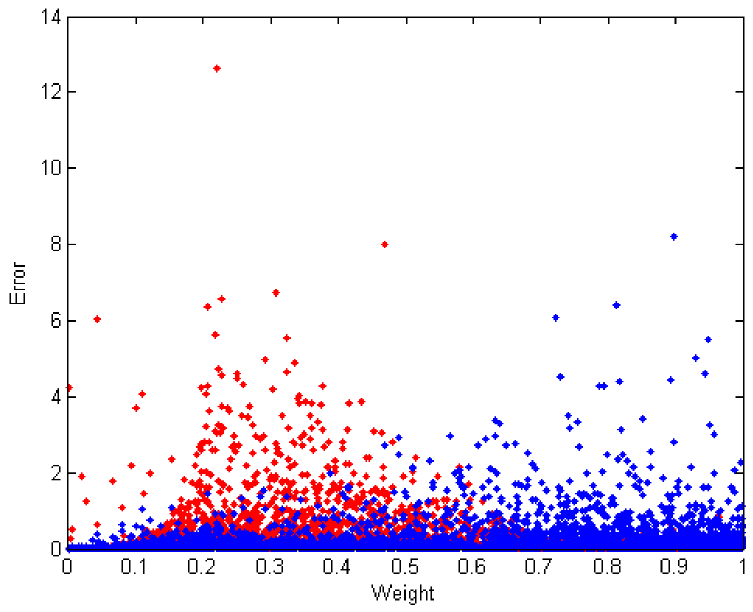

Figure 66: The absolute differences between the true value and the 0.5826-approximation (red) and the absolute differences between the true value and our adjusted 0.5826-approximation (blue) plotted against the weight. The plot was obtained using 5,000 simulated parameter sets.

In Figure 66 it can be seen that, just as in the $g = 0, r = 0$ case (compare Figure 61), our adjustment improves the approximation for small weights (although apparently not as much). For large weights the approximation gets considerably worse using the adjustment, however.

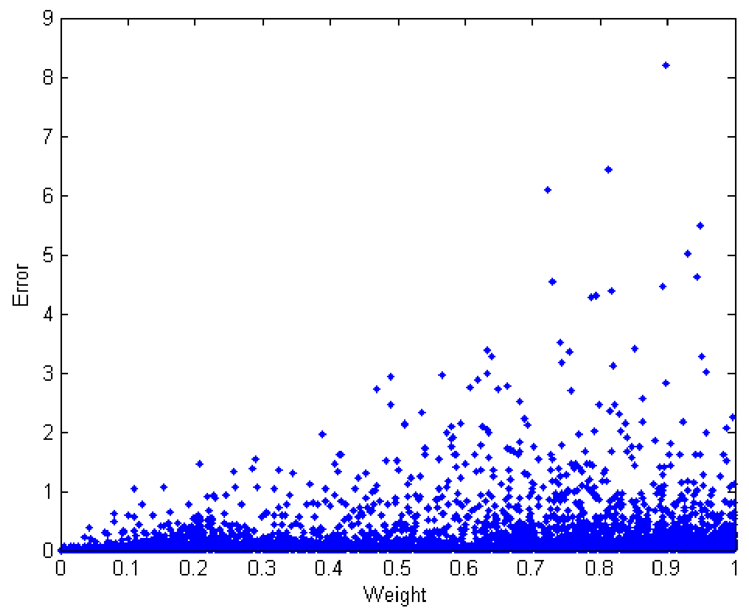

Figure 67: The absolute differences between the true value and our adjusted 0.5826-approximation (blue) plotted against the weight. The plot was obtained using 5,000 simulated parameter sets.

It would have been natural to expect a similar result as in the case with $g = 0$ and $r = 0$. This is not what we get, however, as you can see in Figure 67. We get the largest absolute error from the new adjustment when the weights are large.

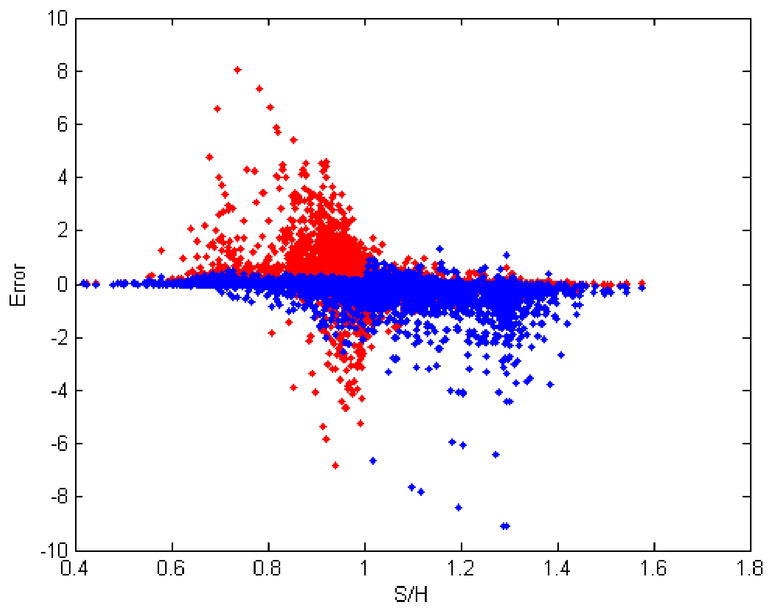

Figure 68: Differences between true values and prices obtained with our adjusted 0.5826-approximation (blue) and differences between true values and 0.5826-approximated prices (red), plotted against $\frac{S}{H}$. 1,000 simulated parameter sets with normally distributed S, $g \neq 0, r \neq 0$.

Just as in the case with $g = 0, r = 0$ the new adjustment improves the approximation of the price significantly when $S/H < 1$. This can be seen in Figure 68. The adjusted approximation even gives several worse prices above this level.

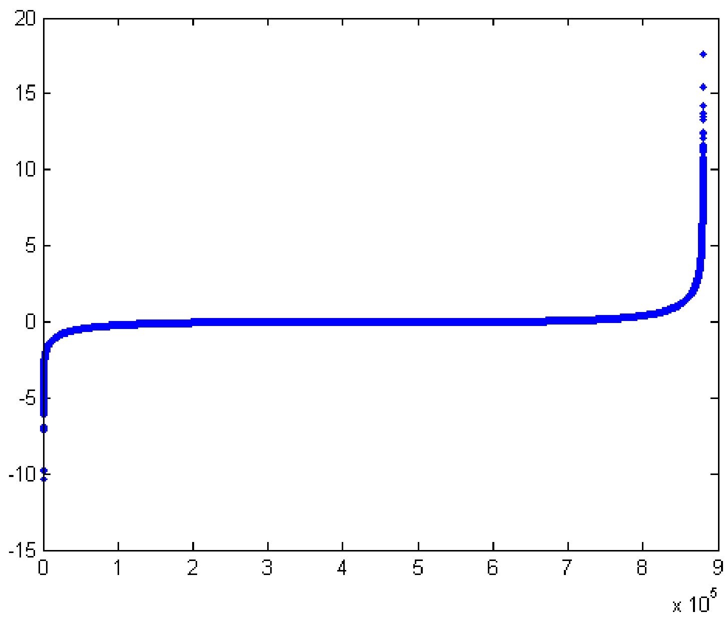

Figure 69: Sorted differences between the absolute differences between the true value and the 0.5826-approximation and the absolute differences between the true value and our adjusted 0.5826-approximation. The plot was obtained using 80,000 simulated parameter sets.

In Figure 69 you can see the differences between the absolute errors obtained with the plain 0.5826-approximation and the absolute errors from the adjusted approximation. The result is great in some cases but it also makes the pricing worse in some.

5.3.2 Second iteration

We are neither satisfied with the percentage of improvements nor the median absolute error from the first iteration. We conclude from for instance Figure 68, however, that our approach performs very well to the left of the barrier. A new idea is therefore to find a level of S/H above which only the plain (unadjusted) 0.5826-approximation will be used. This is the subject of this second iteration.

5.3.2.1 Determination of optimal γ

The analysis in this section follows these steps:

1. State optimization problem.
2. Determine optimal values of γ.

We will not separately determine the optimal number of simulated parameter sets in this step, but rather assume that 20,000 is suitable here as well. The steps are described in turn below.

5.3.2.1.1 Optimization problem

The idea is to minimize the ratio between the percentage of unimprovements (versus the plain 0.5826-approximation) and the average improvement. To achieve this, the optimizer will minimize the numerator and maximize the denominator (which is exactly what we want). Were it not for the *max* functions used in (5.2), the optimizer could get the idea to make the denominator negative and/or to make the numerator zero.

We use this approach to find the optimal limit of S/H, b, above which only the plain 0.5826-approximation should be used, and simultaneously find the optimal value of γ, γ^r. In other words, our approximation adjustment will only be used below b. The minimization problem is

$$\min_{\gamma,b}\left[\frac{max\left(\epsilon,\left(\sum_{i=1}^{n}\sum_{j=1}^{m}1_{e_{O,i,j}\leq e_{N,i,j},S_{i,j}/H_i<b}\right)/\left(\sum_{i=1}^{n}\sum_{j=1}^{m}1_{S_{i,j}/H_i<b}\right)\right)}{\max\left(\epsilon,\overline{\boldsymbol{D_{O,N}}}\right)}\right] over\ all\ \gamma \geq 0, \tag{5.2}$$

where

$$1_{e_{O,i,j}\leq e_{N,i,j},S_{i,j}/H_i<b}=\begin{cases}1, & if \quad e_{O,i,j}\leq e_{N,i,j}\ and\ S_{i,j}/H_i<b\\ 0, & if\ e_{O,i,j}>e_{N,i,j}\ and/or\ S_{i,j}/H_i\geq b,\end{cases}$$

$$1_{S_{i,j}/H_i<b}=\begin{cases}1, & if \quad S_{i,j}/H_i<b\\ 0, & if \quad S_{i,j}/H_i\geq b,\end{cases}$$

$$\epsilon=1\cdot 10^{-3},$$

$$\overline{\boldsymbol{D_{O,N}}}=\overline{\boldsymbol{e_O}-\boldsymbol{e_N}}=\frac{1}{n\cdot m}\sum_{i=1}^{n}\sum_{j=1}^{m}\left(e_{O,i,j}-e_{N,i,j}\right),$$

$$e_{O,i,j}=\left|C_{DO}^{FD}\left(T_i,S_{i,j},X,H_i,\sigma_i,r_i,g_i,n_{monit,i}^{eq},n_{steps}\right)-C_{DO}^{0.5826}\left(T_i,S_{i,j},X,H_i,\sigma_i,r_i,g_i,n_{monit,i}^{eq}\right)\right|,$$

$$e_{N,i,j}=\left|C_{DO}^{FD}\left(T_i,S_{i,j},X,H_i,\sigma_i,r_i,g_i,n_{monit,i}^{eq},n_{steps}\right)-approximation_{i,j}\right|,$$

with

$$\begin{aligned}approximation_{i,j} &= 1_{S_{i,j}/H_i<1}\left(k_{i,j}V_{i,j}^{eq}+\left(1-k_{i,j}\right)V_{i,j}^{first}\right)+(1\\ &\quad -1_{S_{i,j}/H_i<1})C_{DO}^{0.5826}\left(T_i,S_{i,j},X,H_i,\sigma_i,r_i,g_i,n_{monit,i}^{eq}\right),\end{aligned}$$

$$\hat{T}_{digital}=\gamma\Delta t^{first}.$$

$V_{i,j}^{eq}$, $V_{i,j}^{first}$, $k_{i,j}$ and $n_{monit,i}^{eq}$ are defined as in section 5.3.1.1.1. $1_{e_{O,i,j}\leq e_{N,i,j},S_{i,j}/H_i<b}$ and $1_{S_{i,j}/H_i<b}$ are, as is obvious from the definitions, indicator functions.

5.3.2.1.2 Optimal γ

In the second iteration the optimal values of γ and b obtained with r and g both equal to zero are $\gamma^r=1.15$ and $b^r=0.975$, respectively.

We do not perform any tests of these results, but rather conclude that since the optimal value of the limit b is very close to one it is more intuitive to set $b=1$, which is equivalent to only using the adjustment to the left of the barrier. This idea is examined in the next iteration.

5.3.3 Third iteration

In the third iteration we set the limit, b, above which only the plain 0.5826-approximation is to be used, to one. We determine optimal values of γ under these circumstances.

5.3.3.1 Determination of optimal γ

The analysis in this section follows these steps:

1. State optimization problems.
2. Determine optimal number of simulated values.
3. Determine optimal values of γ.

The steps are described in turn below.

5.3.3.1.1 Optimization problems

In all optimization problems in this third iteration we use $b = 1$, and our adjustment is only used for $S/H < b$ (i.e. when the underlying is below the barrier).

In this iteration we use three different optimization approaches. These are the norm approach from section 5.3.1.1.1, a new approach where we maximize the percentage of improvements, and another new approach where the average improvement using our adjusted approximation is maximized.

5.3.3.1.1.1 Optimize norm of weighted errors

The minimization problem from the first iteration has to be adjusted to this new setting, and the adjustment is only done when $S/H < 1$. Indicator functions are used for this. Below is the redefined minimization problem.

$$\min_{\gamma}\left[\sum_{i=1}^{n}\sum_{j=1}^{m}\left|C_{DO}^{FD}\left(T_i,S_{i,j},X,H_i,\sigma_i,r_i,g_i,n_{monit,i}^{eq},n_{steps}\right)-approximation_{i,j}\right|\right] over\ all\ \gamma \geq 0,$$

with

$$\begin{aligned}&approximation_{i,j}\\&\qquad= 1_{S_{i,j}/H_i<1}\left(k_{i,j}V_{i,j}^{eq}+\left(1-k_{i,j}\right)V_{i,j}^{first}\right)\\&\qquad+\left(1-1_{S_{i,j}/H_i<1}\right)C_{DO}^{0.5826}\left(T_i,S_{i,j},X,H_i,\sigma_i,r_i,g_i,n_{monit,i}^{eq}\right),\end{aligned}$$

where $V_{i,j}^{eq}$, $V_{i,j}^{first}$, $k_{i,j}$ and $n_{monit,i}^{eq}$ are defined as in section 5.3.1.1.1, and $1_{S_{i,j}/H_i<b}$ is defined as in section 5.3.2.1.1.

5.3.3.1.1.2 Optimize percentage of improvements

To optimize (i.e. maximize) the percentage of improvements versus the plain 0.5826-approximation, what we practically do is to minimize the percentage of unimprovements by solving

$$\min_{\gamma}\left[\left(\sum_{i=1}^{n}\sum_{j=1}^{m}1_{e_{O,i,j}\leq e_{N,i,j},S_{i,j}/H_i<b}\right)/\left(\sum_{i=1}^{n}\sum_{j=1}^{m}1_{S_{i,j}/H_i<b}\right)\right] over\ all\ \gamma \geq 0.[50]$$

The indicator functions as well as $e_{O,i,j}$ and $e_{N,i,j}$ are specified as in section 5.3.2.1.1.

[50] As you see the nominator will be zero if there are no simulations with $S_{i,j}/H_i < 1$, we will however not get this problem since roughly 50% of our simulated data will be below this level.

5.3.3.1.1.3 Optimize average improvement

In this approach we maximize the average improvement of the absolute errors we get when using our approximation compared to the plain 0.5826-approximation. Maximizing this average improvement is obviously equivalent to minimizing

$$\min_{\gamma}\left[1/max(\epsilon, \overline{\boldsymbol{D_{O,N}}})\right]\ over\ all\ \gamma \geq 0,$$

where ϵ and $\overline{\boldsymbol{D_{O,N}}}$ are defined as in section 5.3.2.1.1. We use the *max* function with ϵ since we do not want the optimizer to make $\overline{\boldsymbol{D_{O,N}}}$ negative.

5.3.3.1.2 Optimal number of simulated values

A similar analysis as for the first problem has been carried out to find the optimal number of simulated parameter sets. We start with the hypothesis that 20,000 simulations is a suitable number also in this second problem. We verify this for γ using the "percentage approach" described in section 5.3.3.1.1.2 for data with normally distributed values of the underlying and $g = 0, r = 0$. It can be seen in Figure 70 that there is no reason not to stick with 20,000 simulations also for the second problem. The obtained γ is only 0.1% smaller for 20,000 simulations than for 100,000.

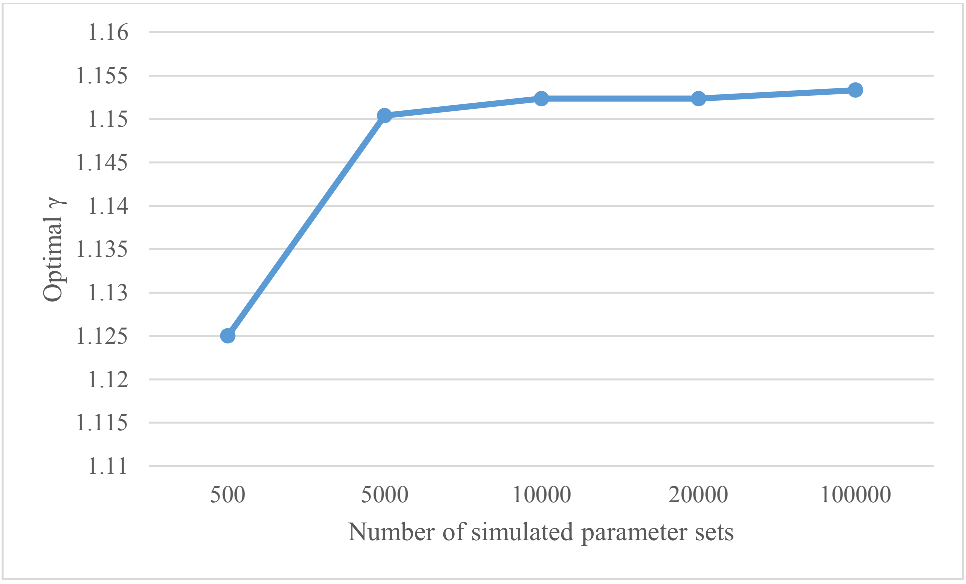

Figure 70: Optimal values of γ for normally distributed S, for different numbers of simulated parameter sets.

5.3.3.1.3 Optimal γ

The optimal value of γ obtained in the minimization of the norm is $\gamma_{g=0,r=0}^{n,b=1} = 1.41$ and $\gamma_{g\neq 0,r\neq 0}^{n,b=1} = 1.57$.

When optimizing the percentage of improvements we obtain $\gamma_{\mathrm{g=0,r=0}}^{p} = 1.15$ and $\gamma_{\mathrm{g\neq 0,r\neq 0}}^{p} = 1.18$.

The optimal values of γ from the optimization approach where we optimize the average improvement are $\gamma_{\mathrm{g=0,r=0}}^{m} = 1.41$ and $\gamma_{\mathrm{g\neq 0,r\neq 0}}^{m} = 1.57$.

5.3.3.2 *Test of optimal γ*

As in the first iteration we test the adjustment based on the different optimal values of γ on data that have not been used to obtain these values. Our adjustment of the 0.5826-approximation is only used when $S/H < 1$, and above the barrier the plain 0.5826-approximation is used. 80,000 simulated parameter sets are used in the test data in all tests (although of course only about half of these are below the barrier). $\gamma_{g=0,r=0}^{n,b=1}$, $\gamma_{\mathrm{g=0,r=0}}^{p}$ and $\gamma_{\mathrm{g=0,r=0}}^{m}$ are tested on data with $r = 0, g = 0$ using $\alpha_{g=0,r=0}^{n}$ and $\gamma_{g\neq0,r\neq0}^{n,b=1}$, $\gamma_{\mathrm{g\neq0,r\neq0}}^{p}$ and $\gamma_{\mathrm{g\neq0,r\neq0}}^{m}$ are tested on data with $r \neq 0$ and $g \neq 0$ using $\alpha_{g\neq0,r\neq0}^{n}$.

5.3.3.2.1 Analysis of $\gamma_{g=0,r=0}^{n,b=1}$

The optimal value of γ obtained in the minimization of the norm is $\gamma_{g=0,r=0}^{n,b=1} = 1.41$. This value is tested in the current section.

Our adjustment improves the pricing in 429,388 cases out of the 440,049 that are below the barrier (97.6%). The corresponding number when a strict improvement is not required is 429,832 (99.7%). The great improvement is illustrated in Figure 71. The reason that many "new" errors are equal to the "old" ones is that only the plain 0.5826-approximation is used above the barrier.

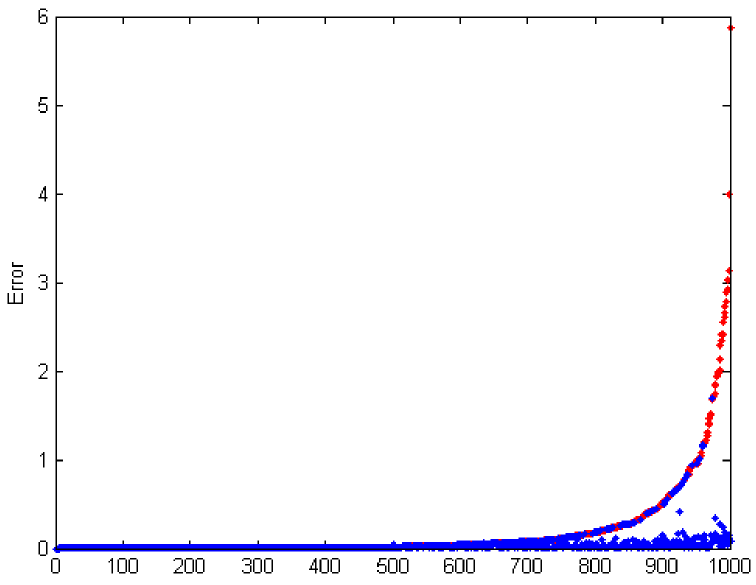

Figure 71: Sorted absolute differences between the true value and the 0.5826-approximation (red) and respective absolute differences between the true value and our adjusted 0.5826-approximation (blue). The plot was obtained using 1,000 simulated parameter sets.

When comparing the approximated prices to the true prices, we get a mean absolute error of 0.0446 (without adjustment: 0.175) and a median of 0.00750 (without adjustment: 0.0142). The average improvement of the absolute errors when using the adjustment compared to just using the 0.5826-approximation is 0.131.[51]

[51] All analysis in this paragraph has been carried out for all the simulated data (i.e. not only below the barrier).

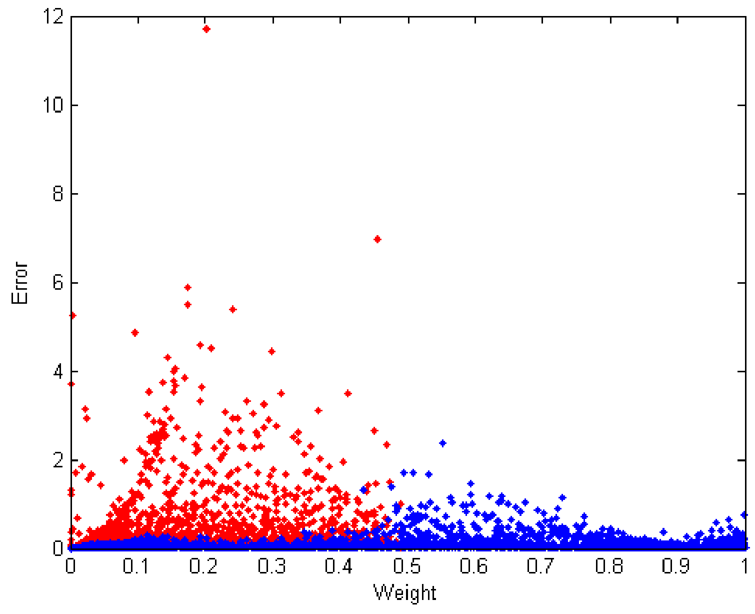

Figure 72: Absolute differences between true values and the 0.5826-approximation (red) and absolute differences between true values and prices obtained using our adjusted 0.5826-approximation (blue) plotted against the weight. The plot was obtained using 5,000 simulated parameter sets.

You can see in Figure 72 that using our adjustment yields a substantial improvement when the weight is small, whereas the same cannot be said for large weights.

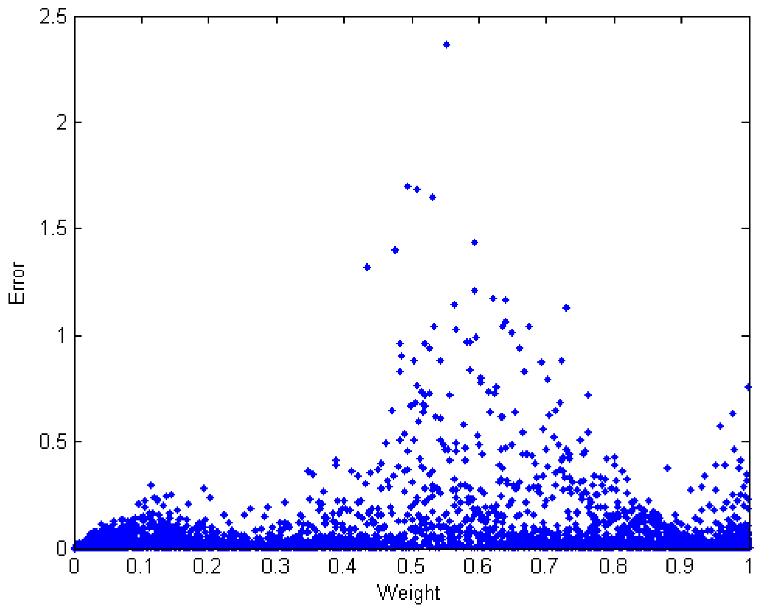

Figure 73: Absolute differences between true values and prices obtained using our adjusted 0.5826-approximation (blue) plotted against the weight. The plot was obtained using 5,000 simulated parameter sets. This is a zoomed in version of Figure 72, but without errors corresponding to the plain 0.5826-approximation.

For the same reasons as in section 5.3.1.2.1 we expected to obtain the largest errors for weights close to 0.5. This is confirmed by Figure 73, although we do notice that errors are in general larger for weights above 0.5 than for weights below that level. Also here most weights are close to zero and one, meaning that the adjustment often uses either the adjustment based on Δt^{first} or the one based on Δt almost exclusively.

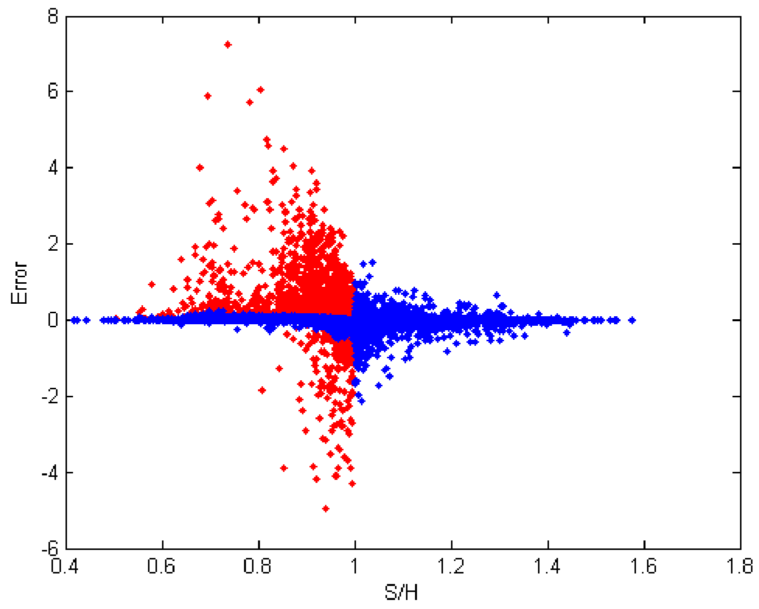

Figure 74: Difference between the true value and the price using our adjusted 0.5826-approximation (blue) and difference between the true value and the 0.5826-approximation (red), plotted against $\frac{S}{H}$. 1,000 simulated parameter sets.

In Figure 74 you can see that our adjustment of the approximation is equal to the 0.5826-approximation when $S/H \geq 1$ since we chose to not use our adjustment term on these. When $S/H < 1$ we get a great improvement of the approximation of the price of the barrier option.

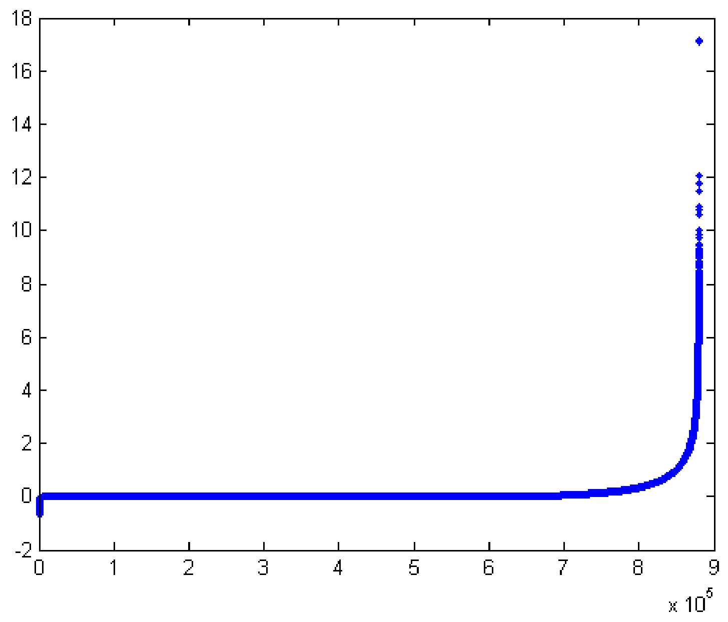

Figure 75: The sorted differences between the absolute differences between the true value and the 0.5826-approximation and the absolute differences between the true value and our adjusted 0.5826-approximation. The plot was obtained using 80,000 simulated parameter sets.

The differences between the absolute errors of the just using the 0.5826-approximation and the absolute errors from the adjusted approximation are displayed in Figure 75. This result is better than the one we get using our adjustment with $\gamma^n_{g=0,r=0}$for all S/H (Figure 64). Now our adjustment makes the pricing approximation worse in considerably fewer cases.

5.3.3.2.2 Analysis of $\gamma_{g\neq 0, r\neq 0}^{n,b=1}$

The optimal value of γ obtained in the maximization of the percentage of improvements is $\gamma_{g\neq 0, r\neq 0}^{n,b=1} = 1.57$. This value is tested in the current section.

As in the $g = 0, r = 0$ case we have 440,049 (50.0%) cases with $S/H < 1$ in the simulated data. We get an improvement with our weighted adjustment in 422,739 out of these cases (96.1%), compare Figure 76. The corresponding number when a strict improvement is not required is 423,821 (96.3%).

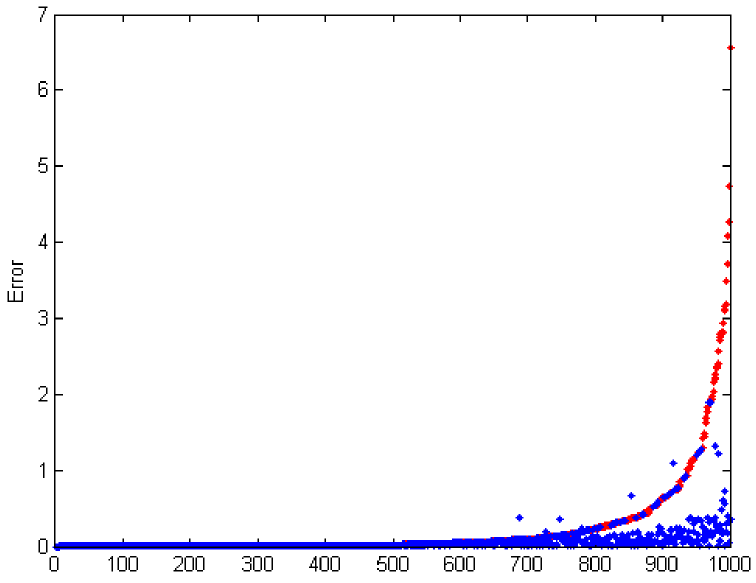

Figure 76: Sorted absolute differences between the true value and the 0.5826-approximation (red) and respective absolute differences between the true value and our adjusted 0.5826-approximation (blue). The plot was obtained using 1,000 simulated parameter sets.

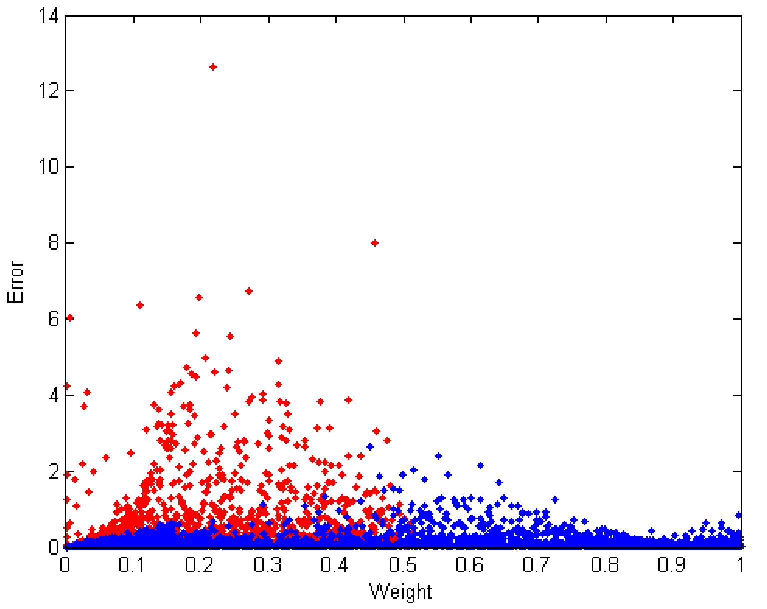

Figure 77: The absolute differences between the true value and the 0.5826-approximation (red) and the absolute differences between the true value and our adjusted 0.5826-approximation (blue) plotted against the weight. The plot was obtained using 5,000 simulated parameter sets.

When we use the weighted adjustment (with $\gamma_{g\neq0,r\neq0}^{n,b=1}$ and $\alpha_{g\neq0,r\neq0}^{n}$) on the 0.5826-approximation we get a mean absolute error of 0.0682 (without adjustment: 0.214) and a median absolute error of 0.0122 (without adjustment: 0.0171). The average improvement of the absolute errors is 0.146.

As we did in the case with $g = 0$ and $r = 0$ we now in the case with simulated g and r want to see how great the weighted adjustment is. In Figure 77 you can easily see that the weighted adjustment improves the result for weights smaller than 0.5.

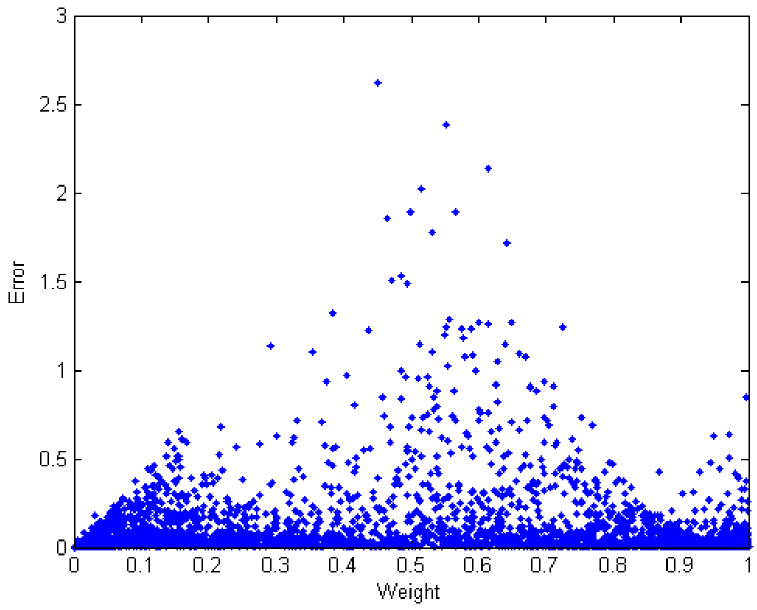

Figure 78: The absolute differences between the true value and our adjusted 0.5826-approximation (blue) plotted against the weight. The plot was obtained using 5,000 simulated parameter sets.

We thought we would get almost the same result as in the case with $g = 0$ and $r = 0$. And we (almost) did, as you can see in Figure 78. We can notice, however, that the errors are more evenly spread out in this case, and that the peak in the middle is more pronounced.

As in the $g = 0, r = 0$ case we see that our adjustment of the approximation is equal to the 0.5826-approximation when $S/H \geq 1$, Figure 79. When $S/H < 1$ we get significant improvements of the approximated barrier option prices.

The differences between the absolute errors of just using the 0.5826-approximation and the absolute errors from the adjusted approximation are displayed in Figure 80. As in the $g = 0, r = 0$ case the result is better than the one we get using our adjustment with $\gamma_{g\neq0,r\neq0}^{n}$ for all S/H (Figure 69). We get a lot less bad adjustments now.

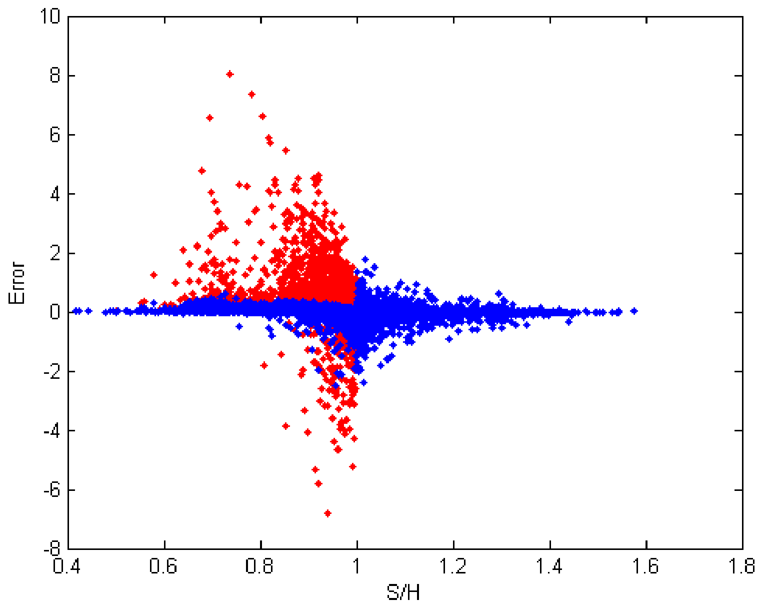

Figure 79: Difference between the true value and the price using our adjusted 0.5826-approximation (blue) and the difference between the true value and the 0.5826-approximation (red), plotted against $\frac{S}{H}$. 1,000 simulated parameter sets.

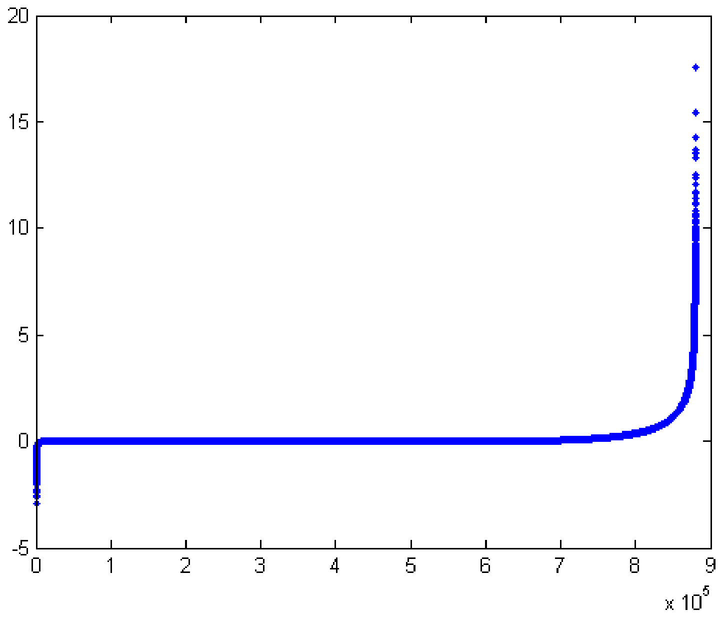

Figure 80: Sorted differences between the absolute differences of the true value and the 0.5826-approximation and the absolute differences between the true value and our adjusted 0.5826-approximation. The plot was obtained using 80,000 simulated parameter sets.

5.3.3.2.3 Analysis of $\gamma^{p}_{g=0,r=0}$

The optimal value of γ obtained in the maximization of the percentage of improvements is $\gamma^{p}_{\mathrm{g=0,r=0}} = 1.15$. This value is tested in the current section.

Our adjustment improves the pricing in 434,594 cases out of the 440,049 that are below the barrier (98.79%). The corresponding number when a strict improvement is not required is 435,071 (98.87%). The dramatic improvement is illustrated in Figure 81. The reason that many “new” errors are equal to the “old” ones is that only the plain 0.5826-approximation is used above the barrier.

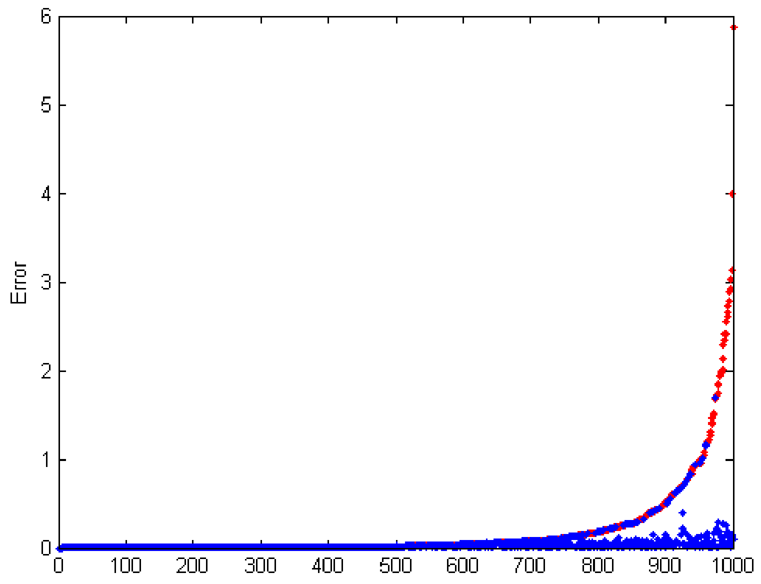

Figure 81: Sorted absolute differences between the true value and the 0.5826-approximation (red) and respective absolute differences between the true value and t our adjusted 0.5826-approximation (blue). The plot was obtained using 1,000 simulated parameter sets.

When comparing the approximated prices to the true prices, we get a mean absolute error of 0.0453 (without adjustment: 0.175) and a median of 0.00790 (without adjustment: 0.0142). The average improvement of the absolute errors is 0.130.[52]

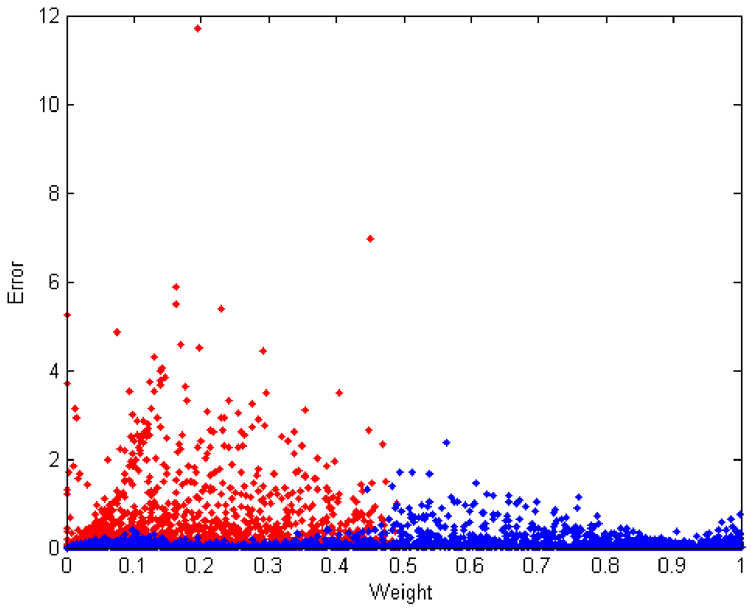

Figure 82: The absolute differences between the true value and the 0.5826-approximation (red) and the absolute differences between the true value and our adjusted 0.5826-approximation (blue) plotted against the weight. The plot was obtained using 5,000 simulated parameter sets.

It is evident from Figure 82 that the same conclusion can be drawn as in the previous cases – the adjusted approximation works way better for weights below approximately 0.5.

[52] All analysis in this paragraph has been carried out for all the simulated data (i.e. not only below the barrier).

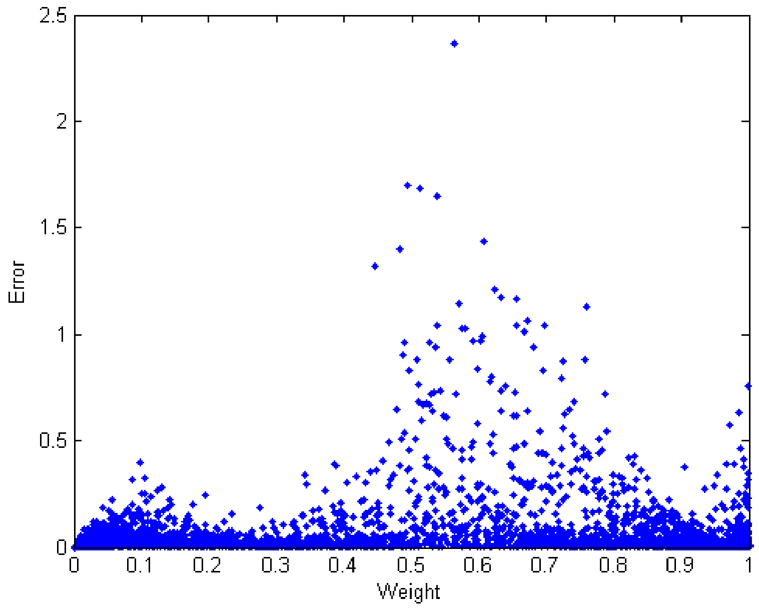

Figure 83: The absolute differences between the true value and our adjusted 0.5826-approximation (blue) plotted against the weight. The plot was obtained using 5,000 simulated parameter sets.

From Figure 83 it can be established that the largest errors obtained with the adjusted approximation are obtained for weights around 0.5, and that errors are generally larger for large weights than for small ones.

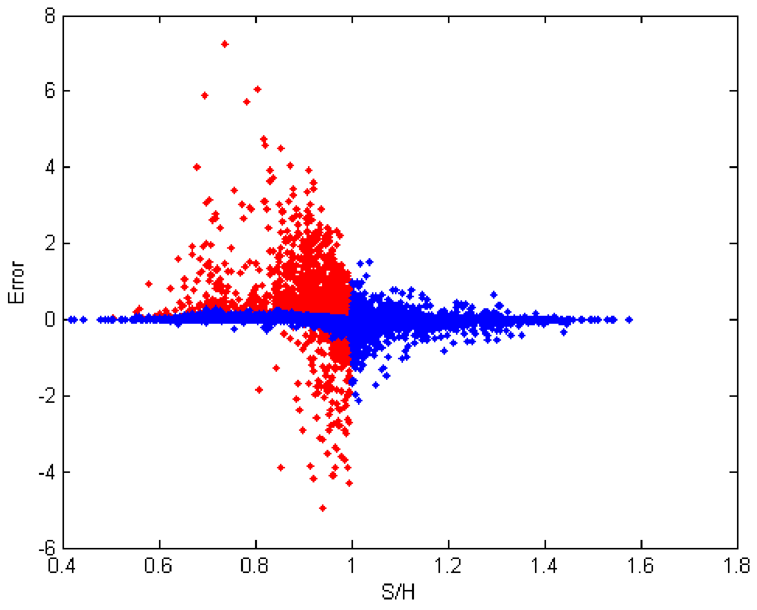

Figure 84: Difference between the true value and the price using our adjusted 0.5826-approximation (blue) and the difference between the true value and the 0.5826-approximation (red), plotted against $\frac{S}{H}$. 1,000 simulated parameter sets.

You can see in Figure 84 that our adjusted approximation is equal to the 0.5826-approximation when $S/H \geq 1$ since we do not use our adjustment term on these. When $S/H < 1$ we get great improvements of the approximated barrier option prices.

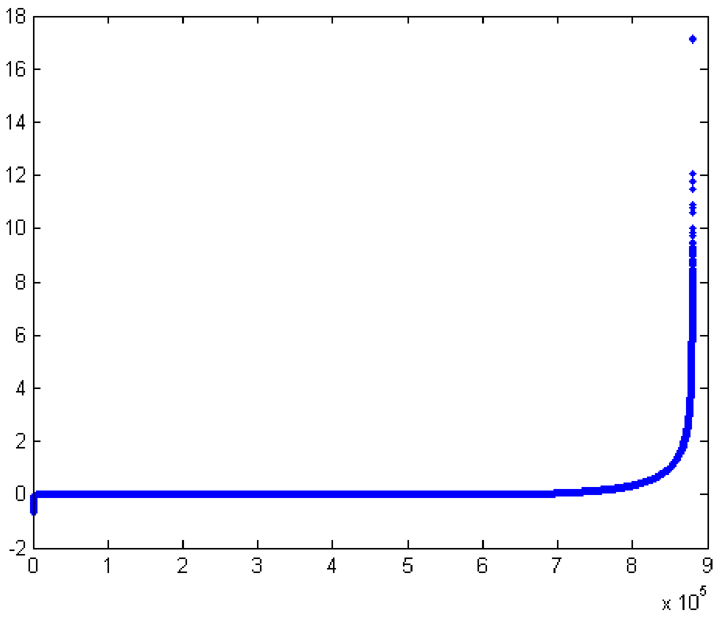

Figure 85: Sorted differences between the absolute differences between the true value and the 0.5826-approximation and the absolute differences between the true value and our adjusted 0.5826-approximation. The plot was obtained using 80,000 simulated parameter sets.

The result in Figure 85 is better than the one we get using our adjustment with $\gamma^{n}_{g=0,r=0}$ for *all* S/H (compare Figure 64). Now our adjustment improves the approximation in almost all cases.

5.3.3.2.4 Analysis of $\gamma^{p}_{\mathrm{g}\neq 0,\mathrm{r}\neq 0}$

The optimal value of γ obtained in the maximization of the percentage of improvements is $\gamma^{p}_{\mathrm{g}\neq 0,\mathrm{r}\neq 0} = 1.18$. This value is tested in the current section.

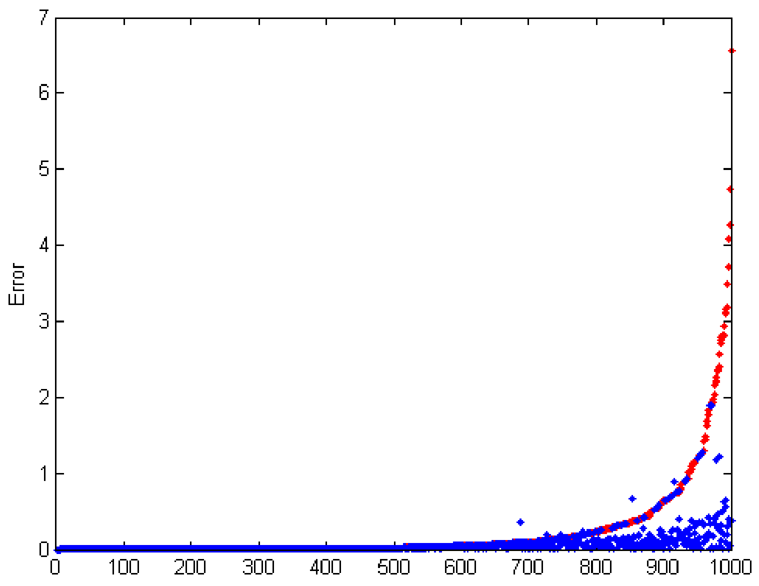

Figure 86: Sorted absolute differences between the true value and the 0.5826-approximation (red) and respective absolute differences between the true value and our adjusted 0.5826-approximation (blue). The plot was obtained using 1,000 simulated parameter sets.

As in the $g = 0, r = 0$ case we have 440,049 (50.0%) cases with $S/H < 1$ in our simulated data. We get an improvement with our weighted adjustment in 431,875 out of these cases (98.1%), compare Figure 86. The corresponding number when a strict improvement is not required is 433,112 (98.4%).

When we use the weighted adjustment (with $\gamma^{p}_{\mathrm{g}\neq 0,\mathrm{r}\neq 0}$ and $\alpha^{\mathrm{n}}_{\mathrm{g}\neq 0,\mathrm{r}\neq 0}$) on the 0.5826-approximation we get a mean absolute error of 0.0695 (without adjustment: 0.214) and a median absolute error of 0.0129 (without adjustment: 0.0171). The average improvement of the absolute errors is 0.145.

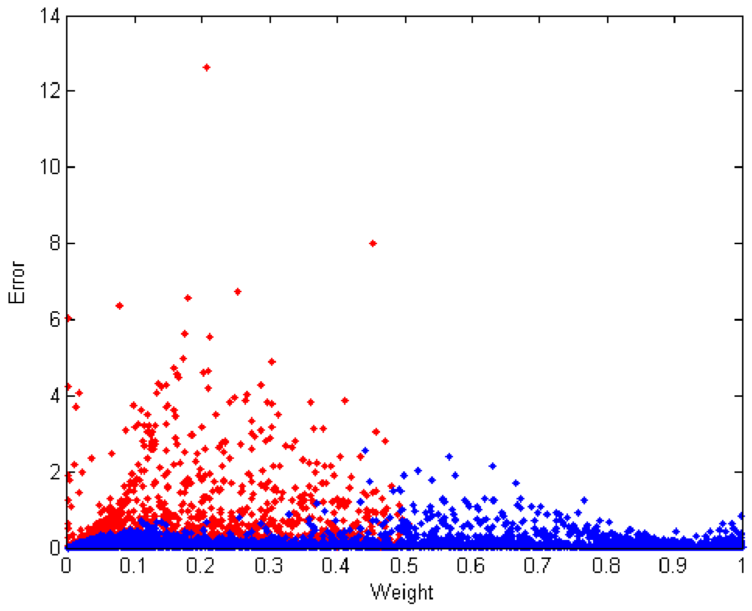

Figure 87: Absolute differences between the true value and the 0.5826-approximation (red) and absolute differences between the true value and our adjusted 0.5826-approximation (blue) plotted against the weight. The plot was obtained using 5,000 simulated parameter sets.

As we did in the case with $g = 0$ and $r = 0$ we now in the case with simulated g and r want to see how well the weighted adjustment performs. You can easily see in Figure 87 that the weighted adjustment improves the result for weights smaller than 0.5.

In Figure 88 it can be seen that the largest errors from the adjusted approximation are once again obtained for weights approximately in the middle.

In Figure 89 we see that, as in the $g = 0, r = 0$ case, our adjustment of the approximation is equal to the 0.5826-approximation when $S/H \geq 1$. When $S/H < 1$ we get significant improvements of the approximated barrier option prices.

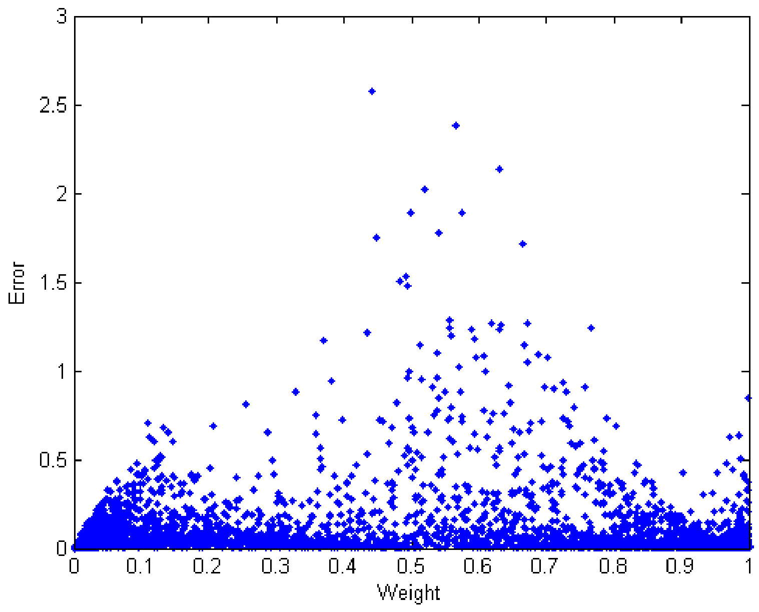

Figure 88: Absolute differences between the true value and our adjusted 0.5826-approximation (blue) plotted against the weight. The plot was obtained using 5,000 simulated parameter sets.

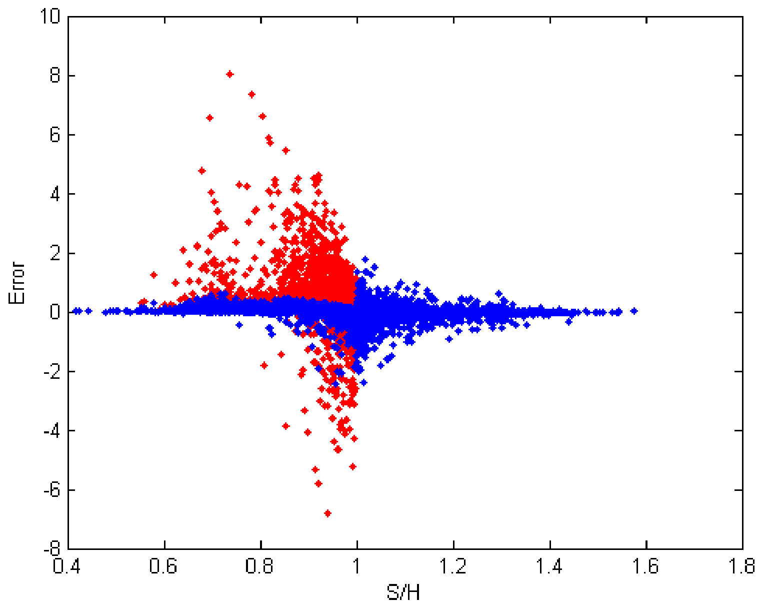

Figure 89: Difference between the true value and our adjusted 0.5826-approximation (blue) and the difference between the true value and the 0.5826-approximation (red), plotted against $\frac{S}{H}$. 1,000 simulated parameter sets.

The differences between the absolute errors of just using the 0.5826-approximation and the absolute errors from the adjusted approximation are displayed in Figure 90. As in the $g = 0$ and $r = 0$ case the result is better when our adjustment is *only* used below the barrier (compare Figure 69).

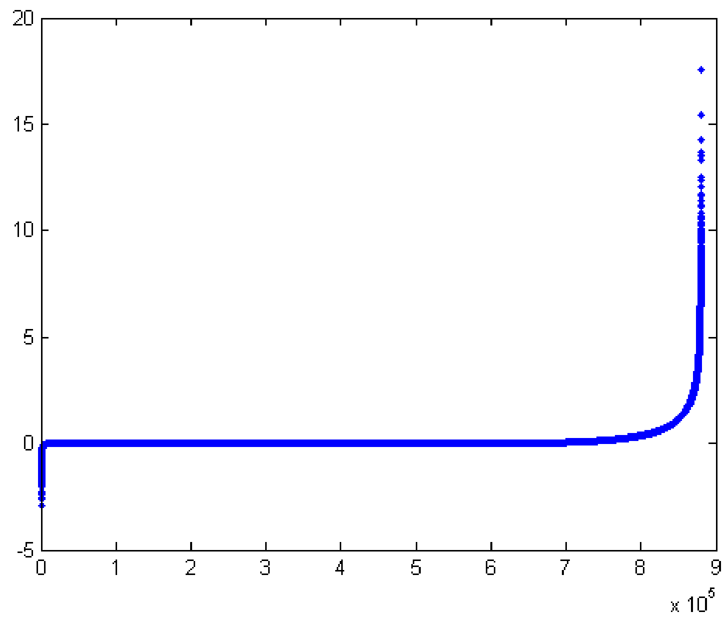

Figure 90: Sorted differences between the absolute differences between the true value and the 0.5826-approximation and the absolute differences between the true value and our adjusted 0.5826-approximation. The plot was obtained using 80,000 simulated parameter sets.

5.3.3.2.5 Analysis of $\gamma^m_{g=0,r=0}$ and $\gamma^m_{g\neq 0,r\neq 0}$

We also optimize the maximum absolute improvement of using the 0.5826-approximation with the adjustment compared to just using the plain 0.5826-approximation. This gives $\gamma^m_{g=0,r=0} = 1.41$ with improvements in 429,388 cases (97.6%) and if we allow equality we get improvements in 429,832 cases (97.7%). The average improvement of the absolute errors is 0.131.

In the case of $g \neq 0, r \neq 0$ we obtain $\gamma^m_{g\neq 0,r\neq 0} = 1.57$ with improvements in 422,739 cases (96.1%) and if we allow equality we get improvements in 423,821 cases (96.3%). The average improvement of the absolute errors is 0.146.

5.4 Conclusion for the second problem

It was at an early stage established that it was preferable to alter the initial hypothesis slightly (specifically, to use the same delta as scale factor in both places). Even then we saw (in the first iteration) that the adjustment performs considerably worse when the underlying is above a certain level, which seemed to be very close to the barrier.

That level was arguably found in the second iteration, and it turned out to be just below, but very close to, the barrier. We decided to only use the adjustment below the barrier, and in the third iteration we optimized γ accordingly (using different methods).

Below we provide the complete suggested approximation formula when $\Delta t^{first} \leq \Delta t$.[53]

$$V = 1_{S/H<1} \cdot \left(kV^{eq} + (1-k)V^{first}\right) + \left(1 - 1_{S/H<1}\right) \cdot C_{DO}^{0.5826}(T,S,X,H,\sigma,r,g,\Delta t),$$

where V^{eq} is the approximated price (using the method introduced in section 4 of this thesis) of a discretely monitored down-and-out call option with time between monitoring instants Δt (and with

[53] The attentive reader may notice that this formula is identical to the one introduced in section *1.3 Conclusion in brief*, but with coefficient values also for the case $g > 0$. It is therefore worth mentioning that we find a separate (although very similar) formula in section 4.4 for the case when $\Delta t^{first} = \Delta t$.

$\Delta t^{first} = \Delta t$). V^{first} is the approximated price using the same method, but with the time between *all* monitoring instants equal to Δt^{first} (although the delta is the same as for V^{eq}). Precise formulation for those variables as well as k and the indicator function are provided below,

$$k = C_{digital}\left(\hat{T}_{digital}, S, \hat{X}_{digital}, \sigma, \hat{r}, g\right),$$

$$V^{eq} = C_{DO}^{0.5826}(T, S, X, H, \sigma, r, g, \Delta t) + \Delta(H_A) \cdot \left(C\left(\hat{T}, S, \hat{X}, \sigma, \hat{r}, g\right) - max\left(S - \hat{X}, 0\right)\right),$$

$$V^{first} = C_{DO}^{0.5826}\left(T, S, X, H, \sigma, r, g, \Delta t^{first}\right) + \Delta(H_A)$$
$$\cdot \left(C\left(\hat{T}^{first}, S, \hat{X}^{first}, \sigma, \hat{r}, g\right) - max\left(S - \hat{X}^{first}, 0\right)\right),$$

where

$$1_{S/H<1} = \begin{cases} 1, & if \quad S/H < 1 \\ 0, & if \quad S/H \geq 1, \end{cases}$$

$$\hat{T}_{digital} = \gamma \Delta t^{first},$$
$$\hat{X}_{digital} = H,$$
$$\hat{r} = 0,$$
$$\hat{T} = \alpha \Delta t,$$
$$H_A = H \cdot e^{-0.5826\sigma\sqrt{\Delta t}},$$
$$H_A^{first} = H \cdot e^{-0.5826\sigma\sqrt{\Delta t^{first}}},$$
$$\hat{X} = H_A,$$
$$\hat{T}^{first} = \alpha \Delta t^{first},$$
$$\hat{X}^{first} = H_A^{first},$$
$$\Delta(H_A) = \frac{\partial C_{DO}^{0.5826}}{\partial S}(H_A).$$

In the case of $g = 0$ we recommend using[54]

$$\alpha = 0.733,$$
$$\gamma = 1.41.$$

The best values of the coefficients obtained in our analyzes in the case of $g > 0$ are

$$\alpha = 0.618,$$
$$\gamma = 1.57.$$

It should just as for the first problem be pointed out that our main result is the one obtained with the cost of carry equal to zero. This is for the same model dependency reasons as in problem one (see for instance section 4.4).

In an extensive test section similar "performance measures" as for the first problem were obtained for optimal values of γ determined with the different methods. The best overall test results in the case of $g = 0$ were obtained essentially using the vector based approach from problem one, but where the adjustment is only used below the barrier. Although certain measures indicate that the "percentage approach" is better when $g > 0$, most measures show also here that the approach from problem one is preferable. That is why we above present the optimal γ obtained with that method. We obtained strict improvements versus the plain 0.5826-approximation in 97.6 and 96.1 percent of the cases when the adjustment was used, for g zero and non-zero respectively. The average improvements of the absolute errors are 0.131 ($g = 0$) and 0.146 ($g > 0$). These average improvements amount to 74.9 and 68.2

[54] These coefficient values have been found to be optimal in the case of zero cost of carry ($g = 0$). We have also found values in the case when g is different from zero. These will be presented in the thesis, but we do not consider those our main results due to additional model dependency introduced in this case.

percent of the mean absolute errors when only using the plain 0.5826-approximation, for the two cases respectively.

6 Summarizing conclusion and further work

This section comprises a brief summarizing conclusion[55] and some suggestions for additional research.

The purpose of this thesis was to improve the (0.5826-)approximation formula presented by Broadie, Glasserman, and Kou in 1997. It can be concluded that our hypotheses were very accurate, although we have altered the hypothesized solution for the second problem a bit. We have found values of the coefficients α and γ that improve the accuracy of the adjusted approximation formula significantly in the tests that we have performed. The accuracy of the approximation is improved in well over 90 percent of the tested cases when our proposed improvements of the 0.5826-approximation are used.

During our work process[56] we have discussed topics which could have been relevant for this thesis, but which we have chosen not to examine further, either because of time limitations or because topics were considered outside our scope. Some of these topics, which we list below, may be interesting subjects of additional research.

- Examine whether our results are applicable to cases with $H > X$.
- Examine to what extent the results we have obtained for down-and-out call options are (with minor alterations) applicable to the pricing of other types of barrier options.
- Examine whether it is possible to obtain even better results with the aid of different types of ansatzes (for both problems one and two). Examples: use ansatzes for other parameters in addition to the time to maturity (such as σ), try non-linear ansatzes.
- Examine the case of non-zero cost of carry in more depth, and in particular the case of a *negative* cost of carry (which we have not examined at all).
- Try to find more theoretical support for our results (or slight alterations of these).

[55] Please refer to the conclusions for the two problems (sections 4.4 and 5.4) for more details.
[56] A few comments on the work process are provided in *Appendix C: Comments on the work process*.

7 References

[1] M. Broadie, P. Glasserman and S. Kou, "A continuity correction for discrete barrier options," *Mathematical Finance,* vol. 7, no. 4, pp. 325-348, October 1997.

[2] L. Clewlow and C. Strickland, Exotic Options - The State of the Art, London: Thomson Business Press, 1997.

[3] S. G. Kou, "On Pricing of Discrete Barrier Options," *Statistica Sinica,* no. 13, pp. 955-964, 2003.

[4] P. Hörfelt, "On the Pricing of Path-Dependent Options and Related Problems," Chalmers University of Technology and Göteborg University, Göteborg, 2003.

[5] D.-H. Ahn, S. Figlewski and B. Gao, "Pricing Discrete Barrier Options with an Adaptive Mesh Model," *The Journal of Derivatives,* vol. Summer, pp. 33-43, 1999.

[6] G. Fusai, I. D. Abrahams and C. Sgarra, "An exact analytical solution for discrete barrier options," *Finance and Stochastics,* no. 10, pp. 1-26, January 2006.

[7] G. Fusai and M. C. Recchioni, "Analysis of quadrature methods for pricing discrete barrier options," *Journal of Economic Dynamics and Control,* vol. 31, no. 3, pp. 826-860, 2007.

[8] E. G. Haug, The Complete Guide to Option Pricing Formulas, 2nd ed., McGraw-Hill.

[9] T. Björk, Arbitrage Theory in Continuous Time, 3rd ed., Oxford University Press, 2009, p. 103.

[10] MathWorks, "Pattern Search Solver - Global Optimization," [Online]. Available: http://www.mathworks.se/products/global-optimization/description6.html. [Accessed 26 April 2014].

[11] MathWorks, "How Pattern Search Polling Works - MATLAB & Simulink - MathWorks Nordic," [Online]. Available: http://www.mathworks.se/help/gads/how-pattern-search-polling-works.html. [Accessed 26 April 2014].

[12] MathWorks, "Pattern Search Options - MATLAB & Simulink - MathWorks Nordic," [Online]. Available: http://www.mathworks.se/help/gads/pattern-search-options.html#f11348. [Accessed 26 April 2014].

[13] MathWorks, "Searching and Polling - MATLAB & Simulink - MathWorks Nordic," [Online]. Available: http://www.mathworks.se/help/gads/searching-and-polling.html. [Accessed 26 April 2014].

[14] A. C. Cameron and P. K. Trivedi, Microeconometrics - Methods and Applications, New York: Cambridge University Press, 2005, pp. 409-410.

8 Appendices

8.1 Appendix A: Notation

This appendix summarizes the main notation used throughout the thesis.

8.1.1 Parameters

8.1.1.1 General

Notation	Formula (when applicable)	Description (when applicable)
C		Price of a vanilla call option
S		Price of underlying asset
X		Strike price
H		Barrier
H_A	$H \cdot e^{-0.5826\sigma\sqrt{\Delta t}}$	0.5826-adjusted barrier using Δt
T		Time to maturity
Δt		Time between monitoring instants
σ		Volatility of underlying asset
g		Cost of carry
r		Risk free interest rate

8.1.1.2 Introduced in problem 1

Notation	Formula (when applicable)	Description (when applicable)
α		Coefficient for Δt in the time to maturity of the vanilla call used in the adjustment term
α^n		Optimal value of α based on the norm
m		Number of simulated values of S in each simulated parameter set
n		Number of simulated parameter sets
n_{monit}^{eq}	$T/\Delta t$	Number of monitoring instants used in V^{eq} (to be defined below)
n_{steps}		Number of steps in finite difference based pricing function
$\hat{X}$	H_A	Strike price of the vanilla call used in the adjustment term
T_{call}		Time to maturity of vanilla call option

8.1.1.3 Introduced in problem 2

Notation	Formula (when applicable)	Description (when applicable)
γ		Coefficient for Δt in the time to maturity of the digital call that is used as weight factor in our adjusted approximation
γ^n		Optimal value of γ based on the norm
γ^p		Optimal value of γ based on percentage of improvements
γ^m		Optimal value of γ based on average improvement between the absolute errors of the approximation and our adjustment of approximation
γ^r		Optimal value of γ based on the ratio between percentage of improvements and the average improvement

n_{monit}		Number of monitoring instants
n_{monit}^{first}	$T/\Delta t^{first}$	Number of monitoring instants used in V^{first} (to be defined below)
H_A^{first}	$H \cdot e^{-0.5826\sigma\sqrt{\Delta t^{first}}}$	0.5826-adjusted barrier using Δt^{first}
$\hat{X}^{first}$	H_A^{first}	Strike price of the vanilla call used in V^{first} (defined below) in the adjustment term
Δt^{first}		Time to first monitoring instant
b		The highest level of S/H for which our adjustment of the approximation is used
b^r		Optimal value of b based on the ratio between percentage of improvements and the average improvement

8.1.2 Functions

8.1.2.1 *Introduced in problem 1*

Notation	**Formula (when applicable)**	**Description (when applicable)**
$C_{DO}^{0.5826}$		The 0.5826-approximated (using Δt in SunGard's internal model) price of the discretely monitored down-and-out call option (possibly with some additional improvements in addition to the 0.5826-approximation)
C_{DO}^{FD}		The "exact" (i.e. benchmark) price of a discretely monitored down-and-out call option, calculated using the SunGard finite differences-based pricing function
$\Delta(x)$	$\frac{\partial C_{DO}^{0.5826}}{\partial S}(x)$	The partial derivative with respect to S ("*delta*") of the 0.5826-approximated (using Δt in SunGard's internal function) price of the discretely monitored down-and-out call option, evaluated at $S = x$
$\hat{T}$	$\alpha\Delta t$	Time to maturity of the vanilla call used in the adjustment term
V^{eq}		Approximated price (using the method introduced in this thesis for solving problem one) of a discretely monitored down-and-out call option with time between monitoring instants Δt
α^l		Linear function of g based on different optimal α's
$\alpha^{polyfit}$		Linear function of g based on Matlab's linear regression (*polyfit*)

8.1.2.2 *Introduced in problem 2*

Notation	**Formula (when applicable)**	**Description (when applicable)**
V		Approximated price of a discretely monitored down-and-out call option with $\Delta t^{first} \neq \Delta t$
V^{one}		Approximated price of a discretely monitored down-and-out call option. Uses the same method as V^{eq}, but with the time between monitoring instants equal to Δt^{first}
V^{first}		Identical to V^{one}, but with the same delta as V^{eq}

$C_{digital}$		Price of a digital call option
$\hat{X}_{digital}$	H	Strike price of the digital call used as weight factor in the adjusted approximation
$\hat{T}^{first}$	$\alpha \Delta t^{first}$	Time to maturity of the vanilla call used in V^{first} (defined below) in the adjustment term
$\hat{T}_{digital}$	$\gamma \Delta t^{first}$	Time to maturity of the digital call used as weight factor in the adjusted approximation

All parameters and variables in bold are in matrix or vector form. If nothing else is stated, these have a number of rows equal to the number of simulations, n.

An asterisk, $*$, that is not in superscript, denotes an element-by-element multiplication[57] of two vectors or matrices (with the same dimensions), that produces a new vector or matrix with those same dimensions.

The optimal α and γ have subscript g and r that are either zero or non-zero depending on whether or not they are calculated from data with simulated g and r. For example $\alpha^{n}_{g=0,r=0}$ is calculated from data with $g = 0$ and $r = 0$. If it is a γ it is calculated with the corresponding α. For example $\gamma^{n}_{g\neq 0,r\neq 0}$ is calculated with $\alpha^{n}_{g\neq 0,r\neq 0}$ from data with simulated g and r. When there is a superscript $b = 1$ the limit of S/H above which our adjustment is not used is equal to one.

[57] An obvious exception is the Matlab code in *Appendix B: Data simulation*.

8.2 Appendix B: Data simulation

In this section we describe how we have simulated the data. More or less all Matlab code that we have used for the simulations is provided. The reason that we make a difference in this respect between this and previous sections is that now the code can be intuitively understood, and almost every line of code actually “adds value”. We think that providing the code can simplify the understanding of the chosen distributions. The code should thus be seen as a complement to the text[58].

We assume different distributions for the parameters. The ones that will be used are uniform, normal, lognormal, and Weibull distributions, see the distribution functions of these below. The choice of distribution has in each case been based on two factors; our aim to obtain suitable values for the tests and economic intuition.

Stochastic variables that follow the distributions mentioned in the preceding paragraph have the following probability density functions respectively:

Uniform distribution (from here on denoted $u(a,b)$):

$$f_X(x;a,b) = \begin{cases} \frac{1}{b-a}, & if\ \ a \le x \le b, \\ 0, & if\ \ x < a \ or \ x > b. \end{cases}$$

Normal distribution (from here on denoted $N(\mu,\sigma^2)$):

$$f_X(x;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Log-normal distribution (from here on denoted $\ln N(\mu,\sigma)$):

$$f_X(x;\mu,\sigma) = \frac{1}{x\sigma\sqrt{2\pi}} e^{-\frac{(\ln x-\mu)^2}{2\sigma^2}}, if\ \ x > 0.$$

Weibull distribution (from here on denoted $W(\lambda,k)$):

$$f_X(x;\lambda,k) = \begin{cases} \frac{k}{\lambda}\left(\frac{x}{\lambda}\right)^{k-1} e^{-(x/\lambda)^k}, & if\ \ x \ge 0, \\ 0, & if\ \ x < 0. \end{cases}$$

It is trivial to obtain a uniform distribution with certain characteristics. The normal distribution is also fairly easy to work with since it is entirely described by the first two moments. The characteristics of the lognormal distribution, on the other hand, are arguably not quite as intuitively linked to its parameters. To minimize this obstacle, we made the following Matlab function[59]:

```
function  [mu, sigma] = lognormalpara(mean, std)
% LOGNORMALPARA  Transform the mean and standard deviation of a lognormal
distribution to the parameters mu and sigma that characterize this
distribution.
%   [MU, SIGMA] = LOGNORMALPARA(MEAN, STD) returns mu and sigma.
sigma = sqrt(log(1+(std/mean)^2));
mu = log(mean)-.5*log(1+(std/mean)^2);
end
```

[58] Please note that while the comments to the code may simplify understanding, they have mainly been written for our own use. They are not necessarily clearly written, grammatically correct, and certainly not carefully edited.

[59] The Weibull distribution is not very intuitive, either, but here we have used an iterative approach where we have varied the parameters until a distribution with desirable characteristics was obtained.

To simulate values of the parameters based on the assumed distributions, we start in each case from values simulated from a uniform distribution, $u(0,1)$. The uniform values are created using Halton sequences (which are described in more detail in section 8.2.1). To obtain values from a normal or lognormal distribution using Halton sequences, we use inverse cumulative distribution functions.

```
n = 1e5; % number of simulated values in each Halton sequence
nbrS = 11; % number of Halton sequences corresponding to S
nbrParaExS = 9; % number of other parameters(5+2(för r)+2(för H))(except S)
nbrHalton = nbrParaExS+nbrS; % number of simulated Halton sequences
H = haltonset(nbrHalton, 'skip', 20);
uniform = net(H, n); % the quasi-random "uniform" numbers
```

8.2.1 Halton sequences

When generating uniformly distributed numbers, the coverage of the sample space becomes better if quasi-random numbers are used instead of pseudo-random numbers [14] (such as those generated with the function *rand* in Matlab). Quasi-random numbers are simulations of systematically drawn numbers. The next draw has a tendency to fill the empty space left from the previous observations when using Halton sequences.

Low correlation between the simulated data sequences is also desirable. To further compare Matlab's pseudo-random method to its Halton sequences, sequences consisting of 10^6 numbers are simulated using both Matlab's pseudo-random and Halton methods. Then the absolute sums of their respective correlation matrices are computed and compared. The Halton method generally performs better in this respect. This supports the decision to use Halton sequences in the thesis.

Explaining how Halton sequences are calculated is easiest done by providing an example[60]. When simulating a Halton sequence you have to choose a prime number as base (nonprime numbers as base will provide a non-unique Halton sequence). In this example the prime number 2 is used. The unit interval, (0, 1), is first divided into two parts. Since we use the number 2 as base in this example, our dividing point is $1/2$. The next step is to divide each part into two new parts, dividing points 1/4 and $3/4$. These three points become the first elements in the Halton sequence. Continuing by dividing the four parts into two equally sized (in our example) parts and so on gives the Halton sequence $\{1/2, 1/4, 3/4, 1/8, 3/8, 5/8, \dots\}$. Using the prime number 3 as base instead would give the following Halton sequence: $\{1/3, 2/3, 1/9, 2/9, 4/9, 5/9, \dots\}$.

8.2.2 Parameter simulation for problem 1

The simulation procedures for the different parameters are described in turn below.

8.2.2.1 Strike, X

We use a fixed strike of $X = 100$ for all simulations.

```
%% X - strike price
X = 100; % Strike
```

8.2.2.2 Cost of carry, g

The cost of carry, g, should be less than r (which is described in the next section), around 0-5% (but with some larger values) and have the highest density at 2-4%. We fulfill these conditions by using a Weibull distribution, $W(0.036, 2.6)$.

[60] This example is heavily inspired by [14].

```
lambda = 0.036; k = 2.6;
halton_G = uniform(:,1);
g = wblinv(halton_G, lambda, k); % Weibull distributed
```

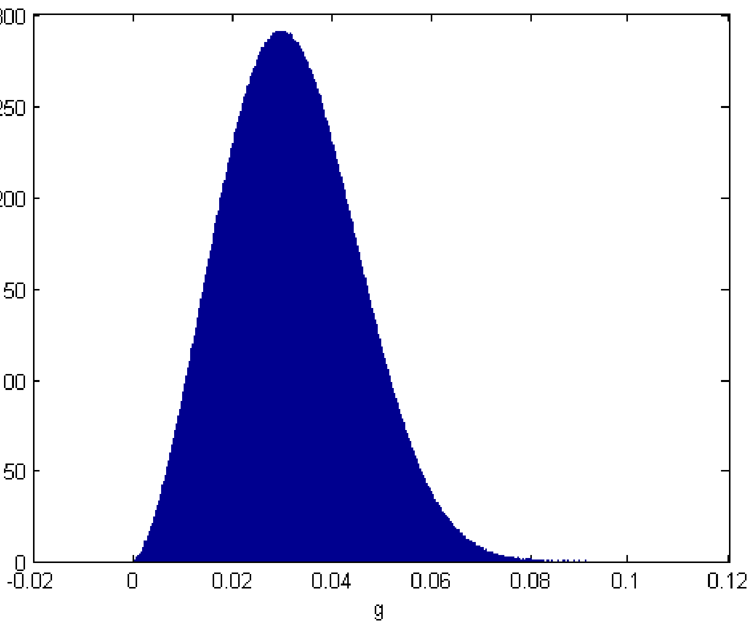

Figure 91: Histogram for g.

8.2.2.3 Risk free rate of interest, r

We want an interest rate, r, that is somewhat uniformly distributed (for each parameter set) between the cost of carry, g, and 5%, but with some higher interest rates. When $g \leq 5\%$ we let 95% of the r's be uniformly distributed, $u(g, 0.05)$. The remaining 5% are obtained as $0.05 + 0.005 \cdot Y$, where $Y \sim W(1,1)$. The r's that correspond to values of $g > 5\%$ are obtained as $g + 0.005 \cdot Y$.

```
halton_R = uniform(:,2);
halton_Rand = uniform(:,3);
r=zeros(n,1);
for l = 1:n
    if g(l)<=0.05 % The ones with g<=0.05
        if halton_Rand(l)<=0.95 % 95% of the ones below 0.05 will be uni-
dist between b and 0.05
            raMax = 0.05; % upper limit
            raMin = g(l); % lower limit
            raMax_adj = raMax-raMin;
            r(l) = raMax_adj*halton_R(l)+raMin;
        else % 5% will be "single higher cases".
            r(l) = 0.05+0.005*wblinv(halton_R(l),1,1);
        end
    else % The ones with g>0.05
        r(l) = g(l)+0.005*wblinv(halton_R(l),1,1);
    end
end
```

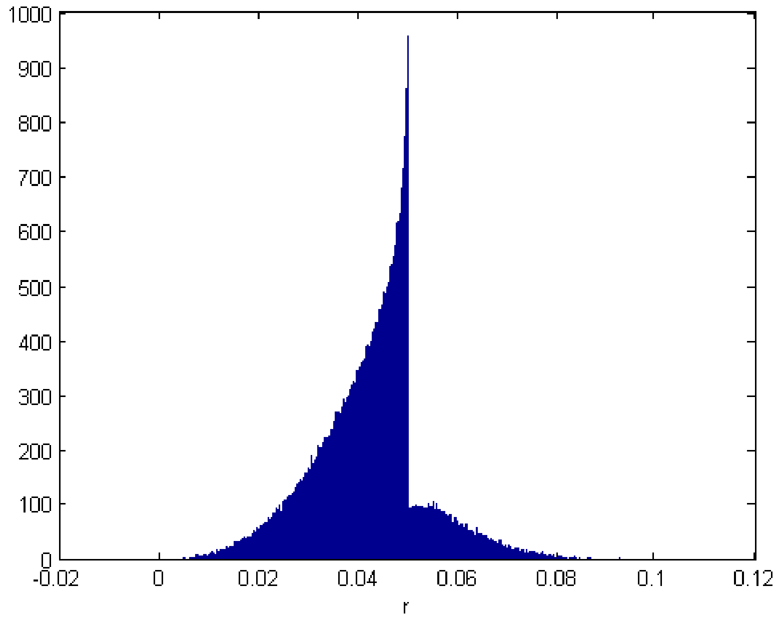

Figure 92: Histogram for r.

8.2.2.4 Standard deviation, σ

We let σ follow a lognormal distribution, $\ln N(-1.2567{,}0.3246)$, which corresponds to an expected value of 0.3 and a standard deviation of 0.1. That way the most common value of σ is approximately 0.25 and reasonably common values are in the interval [0.1, 0.6]. Furthermore, the probability of negative standard deviations is zero.

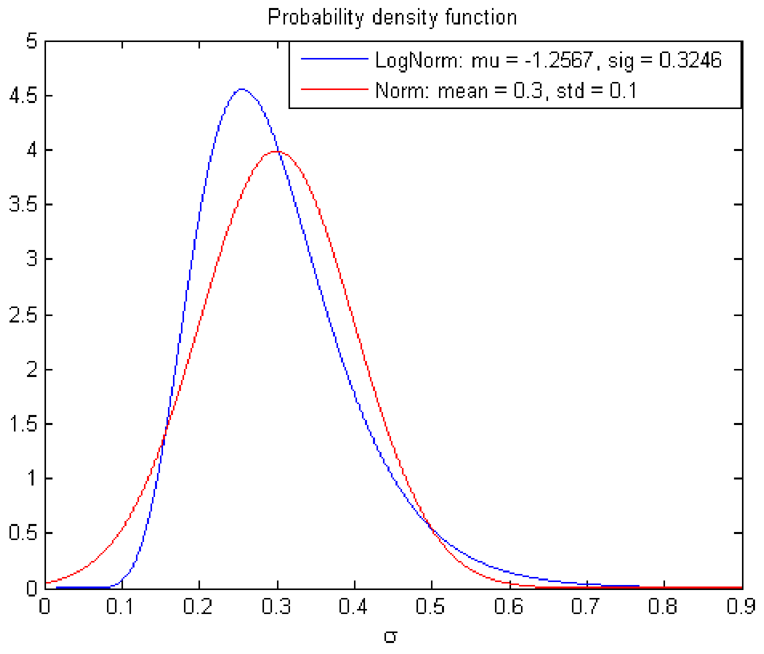

Figure 93: Probability density function for σ.

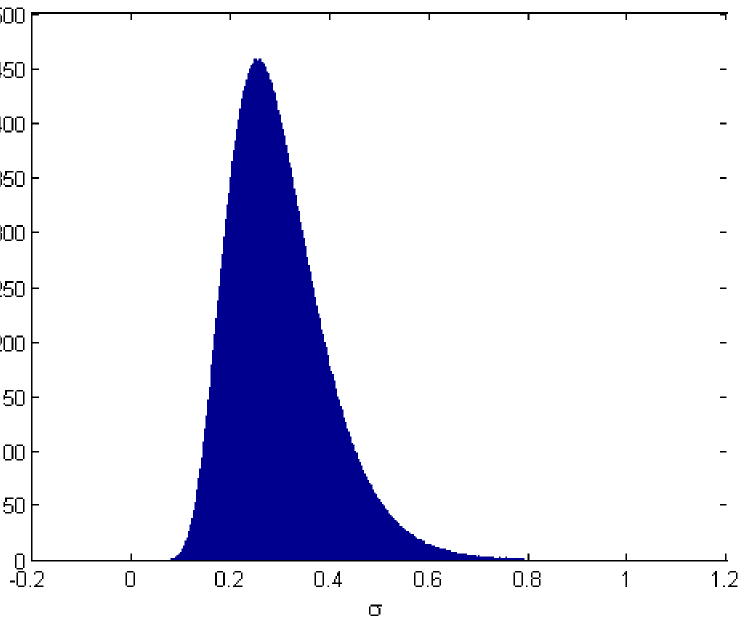

Figure 94: Histogram for σ.

```
%% sigma - standard deviation of the underlying asset
halton_sigma = uniform(:, 4);
exp_val = 0.3; std = 0.1;
[sigma_mu, sigma_sig] = lognormalpara(exp_val, std); % obtain lognormal
parameters corresponding to a certain mean and standard deviation
sigma = logninv(halton_sigma, sigma_mu, sigma_sig); % use the inverse
lognormal cdf on the Halton sequence to obtain the lognormal sigmas
```

8.2.2.5 Time to maturity, T, time between monitoring instants, Δt, and number of monitoring instants, n_{monit}

We want the simulated times to maturity to be in the interval [3 days, 10 years]. Furthermore, we want most of the simulated values to be in the interval [3 months, 2 years]. We decide to let T follow a lognormal distribution, ln$N(0.3466, 0.8326)$, corresponding to both mean and standard deviation equal to 2.

To make sure that we do not obtain values outside the specified interval, we adjust the Halton sequence corresponding to T with the aid of the cumulative distribution function of the specified lognormal distribution. We examine what values of this cdf correspond to 3 days and 10 years, respectively. Then we make sure that the adjusted Halton sequence only contains probabilities within the interval [3 days, 10 years]. The T we get now is our “temporary” T. The final T is very similar but adjusted to fit the n_{monit}^{eq} and Δt. The adjustment methodology is easily seen in the code below.

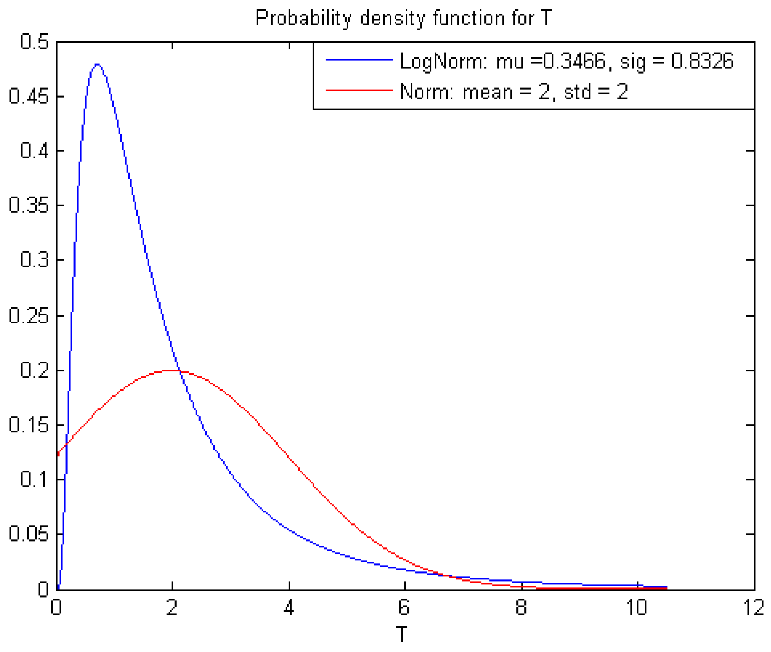

Figure 95: Unadjusted probability density function for T (blue), and "corresponding" normal pdf (red).

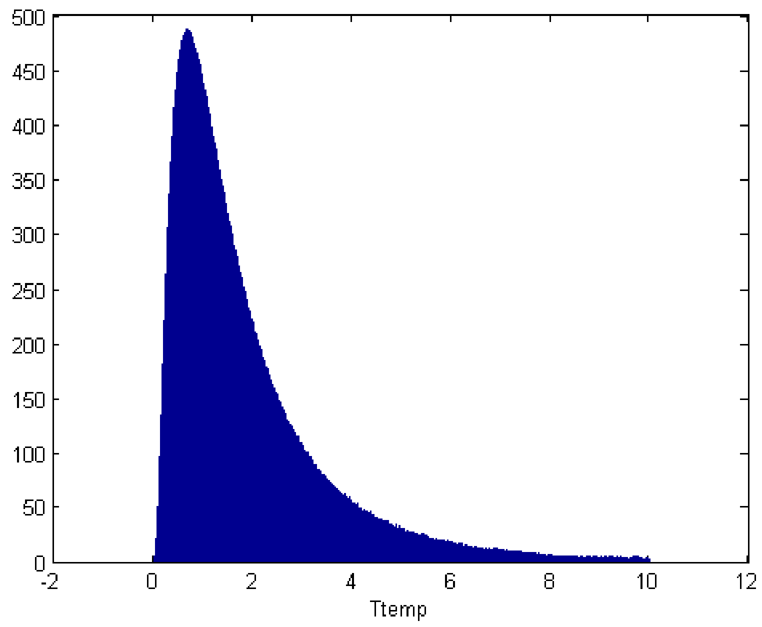

Figure 96: Histogram for Ttemp.

```
%% T - time to maturity

meanT = 2; stdT = 2;
[muT, sigT] = lognormalpara(meanT, stdT);
haltMax = logncdf(10,muT,sigT); % upper limit 10 years
haltMin = logncdf(3/365,muT,sigT); % lower limit 3 days
haltMax_adj = haltMax-haltMin;
halton_T = haltMax_adj*uniform(:, 5)+haltMin; % adjust Halton sequence to
generate values within interval
Ttemp = logninv(halton_T, muT, sigT);
```

Temporarily moving on to the simulation of the time between monitoring instants, Δt, we allow the following values:

$$\Delta t \in \{1/365, 1/252, 1/52, 2/52, 1/12, 1/6, 1/4, 1/2, 1\}, \tag{8.1}$$

where all values are in years. The only value of Δt that may not be immediately understood is 1/252. The rationale is that a year is commonly assumed to have 252 trading days.

It should be noted that we do *not* allow larger values of Δt than of T.

One might expect a histogram of our simulated Δt to "look uniformly distributed". This is however not completely the case since Δt depends on T. Δt is however uniformly distributed in each parameter set between the different possible Δt (the ones in (8.1) that are between zero and the "temporary" T of the parameter set).

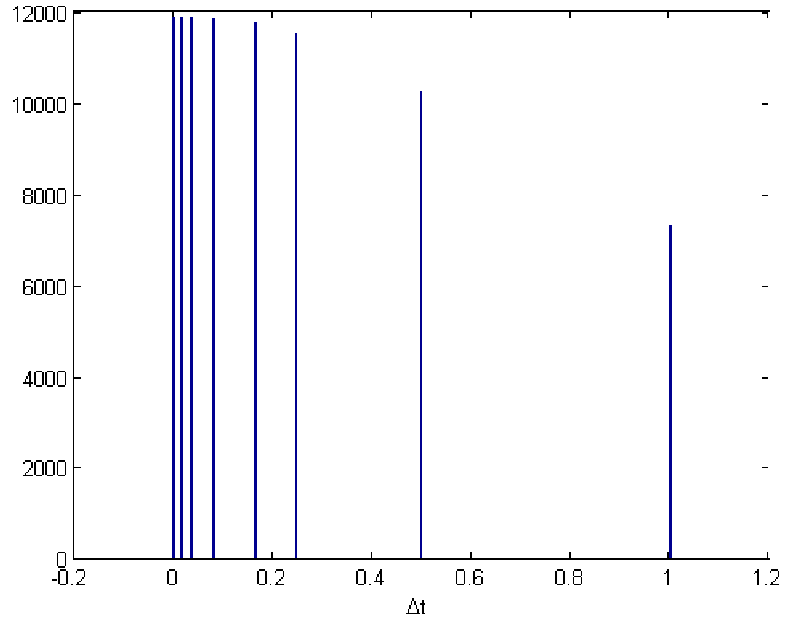

Figure 97: Histogram for Δt.

```
% dt - time interval between monitoring dates
possible_dt = [1/365 1/252 1/52 2/52 1/12 1/6 1/4 1/2 1]; % the possible
intervals
halton_dt = uniform(:, 6);
dt = zeros(n, 1);
A = [1 2 3 4 5 6 7 8 9]; % to get boundaries later
nMonit=zeros(n, 1);
T=zeros(n,1);
for i = 1:n
    allowed_dt = possible_dt <= Ttemp(i); % returns vector with ones in
positions where true, zero elsewhere. Necessary because intervals between
monitoring dates must be smaller than time to maturity.
    number_allowed = sum(allowed_dt,2);
    interval_length = 1/number_allowed; % interval length between
boundaries
    boundaries = A.*interval_length;
    boundaries = boundaries(1:number_allowed); % obtain the boundaries that
should be included.
    current_index = min(nonzeros(find(boundaries >= halton_dt(i)))); %
index of the boundary that is just above the value from the Halton sequence
```

```
    dt(i) = possible_dt(current_index); % Final dt.[61]
    nMonit(i)=round(Ttemp(i)/dt(i));  % We need number of monitoring points
and nMonit*dt should be equal to T.
    T(i)=nMonit(i)*dt(i);
end
```

As is evident from the code above, and as we previously mentioned, we first obtain a "temporary" value of the time to maturity, T. We then use this to obtain the number of monitoring points, n_{monit}^{eq}, by dividing the "temporary" T by the corresponding Δt of the parameter set. n_{monit}^{eq} is rounded to closest integer for each barrier option and the time to maturity is then calculated as $T = n_{monit}^{eq} \cdot \Delta t$.

It should be pointed out that the approach is slightly more involved when $\Delta t^{first} < \Delta t$, compare section 8.2.3.

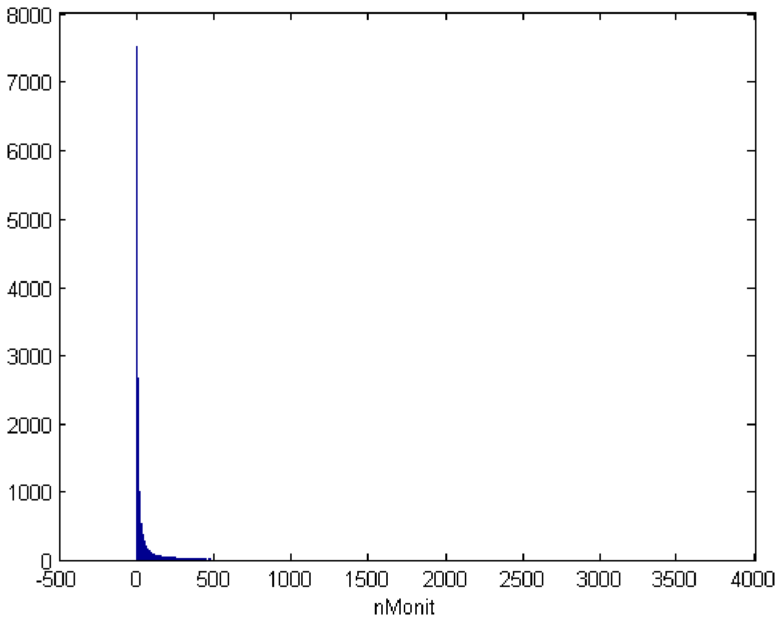

Figure 98: Histogram for n^{monit}.

[61] The final Δt, this is where the simulation for problem 2 will take the Δt and $Ttemp$.

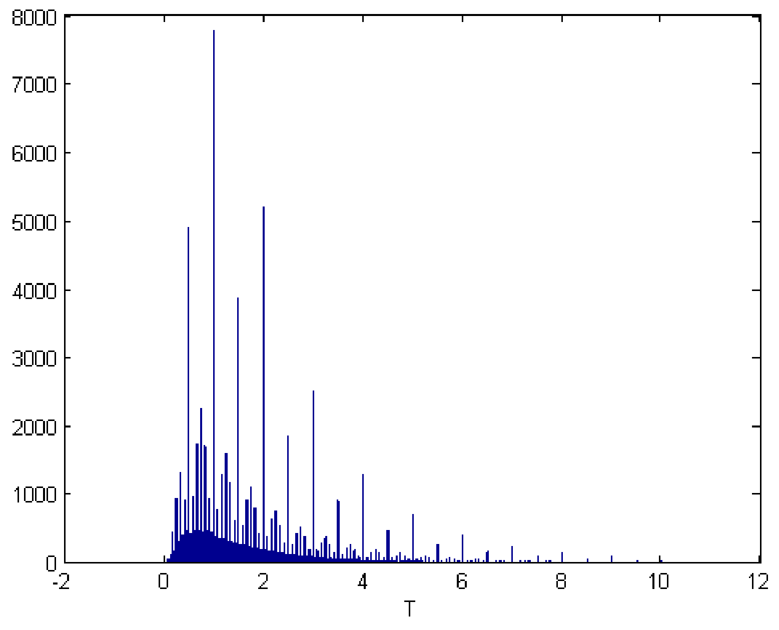

Figure 99: Histogram for T.

8.2.2.6 Barrier, H

Our starting point when specifying the criteria that the distribution for the barrier should fulfill has been suggestions from our supervisor at SunGard. The criteria are:

- The majority of the simulated barrier values should be close to the strike.
- A disproportionally large fraction of the simulated values should be exactly equal to the strike.
- No values of the barrier should be very close to, but not equal to, the strike. The reason is that this is apparently uncommon in the market.
- No values of the barrier should be above the strike (since we do not examine such options in the thesis).
- The lowest possible values of the barrier should not be extremely far away from the strike.

For the minimum H, H_{min}, we want the probability under the risk neutral measure of the underlying going from this level to the strike level during the life time of the option to equal 10%. We think that this renders minimum values of the barrier that are at a suitable distance from the strike. The derivation of H_{min} follows below.

We start as usual from the SDE describing the underlying under the risk neutral measure Q in the Black-Scholes framework,

$$dS_t = gS_t dt + \sigma S_t dW_t.$$

We solve this SDE (given the initial value S_0) to obtain

$$S_T = S_0 e^{\left(g-\frac{\sigma^2}{2}\right)T+\sigma W_T}.$$

On a side note, this also gives the following variance and expected value of the log returns, $\text{Ʀ}_i = \log(\frac{S_{t_i}}{S_{t_{i-1}}})$, where $t_i = t_{i-1} + \Delta t$.

$$E[\text{Ʀ}_i] = \left(g - \frac{\sigma^2}{2}\right)\Delta t,$$
$$Var[\text{Ʀ}_i] = \sigma^2 \Delta t.$$

Since the Ɽ_i are independent, we obtain an annual standard deviation of the log returns of

$$\sqrt{Var\left[\sum_{i=1}^{\frac{1}{\Delta t}} \text{Ɽ}_i\right]} = \sqrt{\sum_{i=1}^{\frac{1}{\Delta t}} Var[\text{Ɽ}_i]} = \sqrt{\frac{1}{\Delta t} Var[\text{Ɽ}_i]} = \sigma.$$

We have $g = 0$ in this initial case, and we further have $S_0 = H_{min}$, $\tilde{X} = 98$. We want to solve

$$Q\big(\tilde{X} \leq S_T | S_0 = H_{min}\big) = 0.1 \Rightarrow$$

$$Q\left(\tilde{X} \leq H_{min} e^{\left(g - \frac{\sigma^2}{2}\right)T + \sigma W_T}\right) = 0.1.$$

This gives

$$Q\left(\frac{\log\left(\frac{\tilde{X}}{H_{min}}\right) - \left(g - \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}} \leq G\right) = 0.1 \Rightarrow$$

$$1 - \Phi\left((\log\left(\frac{\tilde{X}}{H_{min}}\right) - \left(g - \frac{\sigma^2}{2}\right)T)/(\sigma\sqrt{T})\right) = 0.1 \Rightarrow$$

$$\log\left(\frac{\tilde{X}}{H_{min}}\right) - \left(g - \frac{\sigma^2}{2}\right)T = \Phi^{-1}(1 - 0.1)\sigma\sqrt{T} \Rightarrow$$

$$H_{min} = \frac{\tilde{X}}{e^{\Phi^{-1}(1-0.1)\sigma\sqrt{T} + \left(g - \frac{\sigma^2}{2}\right)T}}.$$

Φ denotes the standard normal cumulative distribution function, Q the risk neutral probability measure, and $G \sim N(0, 1)$. To be certain that H_{min} is in fact always *below* (rather than above) $\hat{X}$ that we use the Matlab *max* function to make sure that the exponential in the denominator is larger than one. We make 4% of the barriers equal to the strike ($X = 100$). The rest of the barriers follow an adjusted mirrored log-normal distribution[62] between H_{min} and 98. This is obtained according to the following steps:

1. Obtain values of the cdf (of the distribution in Footnote 62) corresponding to the "boundaries" we want to impose on the distribution, 0.04 and 1.
2. Adjust the Halton sequence with the aid of the boundaries from step one.
3. Subtract the lower boundary from all values of the adjusted distribution to shift it to zero.
4. Divide all values by $(1 - 0.04)$ to make the distribution take all values between zero and one.
5. Subtract all values by one and take the absolute value to mirror it around the y-axis.
6. Multiply by $(98 - H_{min})$ to obtain the correct scale.
7. Add H_{min} to shift the distribution to the desired interval.

```
X0 = 98; %2 percent from 100.
H = zeros(n,1);
minH = X0./(exp(max(norminv(1-0.1,0,1).*sigma.*sqrt(T)+(g-
sigma.^2/2).*T,0)));
muHaltH = -0.80;
sigHaltH = 1.127;
minHaltH = 0.04;
maxHaltH = 1;
maxH = X0;
h_adj = maxH-minH;
```

[62] The unadjusted distribution is $\ln N(-0.8, 1.127)$.

```
haltHLogMax = logncdf(maxHaltH, muHaltH, sigHaltH);
haltHLogMin = logncdf(minHaltH, muHaltH, sigHaltH);
haltHLogMax_adj = (haltHLogMax-haltHLogMin);
halton_H = haltHLogMax_adj.*uniform(:,8) + haltHLogMin;
for i=1:n
    if uniform(i,9)>=0.96 % 4 percent will be equal to Strike.
        H(i) = X;
    else % 96 percent Will be mirrored lognormal distributed from 0 to x0.
        H(i) = h_adj(i)*abs((logninv(halton_H(i), muHaltH, sigHaltH)-
minHaltH)*maxHaltH/(1-minHaltH)-1)+minH(i);
    end
end
```

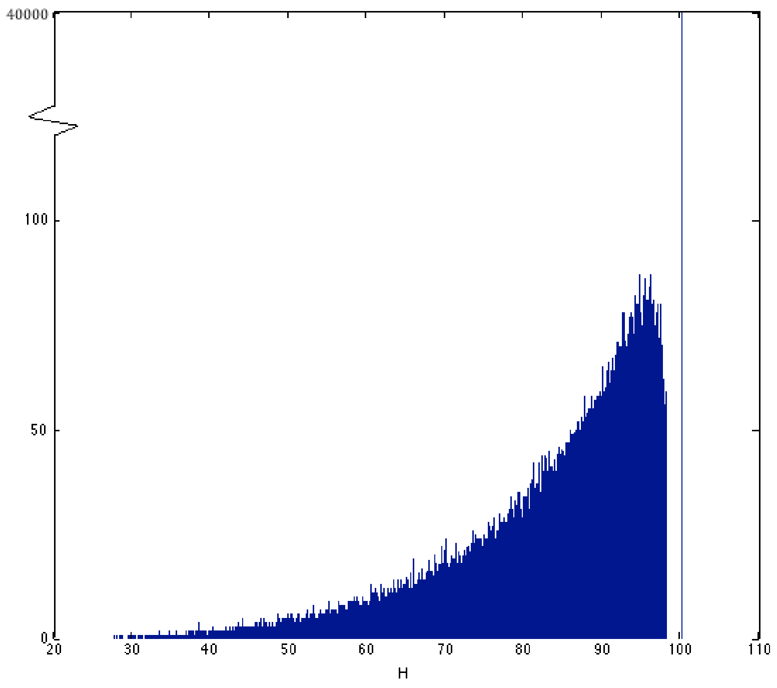

Figure 100: The distribution of H in the call case where H is between H_{min} and X with high density around 90-98 and there is a chance that H is equal to X but zero probability of H∈]98,100[.

8.2.2.7 Underlying asset, S

Our goals when choosing distributions for S are the following:

1. We want the S_j to be symmetric around the adjusted barrier.
2. We want the majority of the S_j to be close to the adjusted barrier, since the adjustment only has a substantial value there.
3. We want the S_j to be more spread out when the standard deviation (of R_i) is high.
4. We want to reflect the increased probability (all else equal) of large movements in the underlying during the option's life time when the time to maturity is large (through more spread out S_j).
5. We do not want negative values of the underlying.
6. We want to have a certain probability of getting simulated S *anywhere* within a reasonable distance of the adjusted barrier.

We have simulated the values of the underlying in two slightly different ways, with distributions that will be referred to as symmetric and normal, respectively. These will be described in turn below.

8.2.2.7.1 Symmetrically distributed S

We obtain the different S_j as $S_j = H_A\left(1 + k_j\sqrt{Var[\text{Ʀ}_i]}\right)$, where Ʀ_i denote the log returns as defined in section 8.2.2.6, and accordingly $Var[\text{Ʀ}_i] = \sigma^2\Delta t$. We want the values of the underlying to be more spread out when the standard deviation is higher, and in this way that is achieved. The k's that we use are $k = \{-2.4925, -2, -1.5075, -1.015, -0.03, 0, 0.03, 1.015, 1.5075, 2, 2.4925\}$.

We can conclude that this distribution meets goals one, three, and five. It also meets goal two to a relatively high extent. The fifth goal is not explicitly considered, in any other way than that a large value of Δt will, all else equal, imply more spread out S_j, and that such a value of Δt will of course not be possible with a very short time to maturity. The sixth goal is obviously not fulfilled with this distribution.

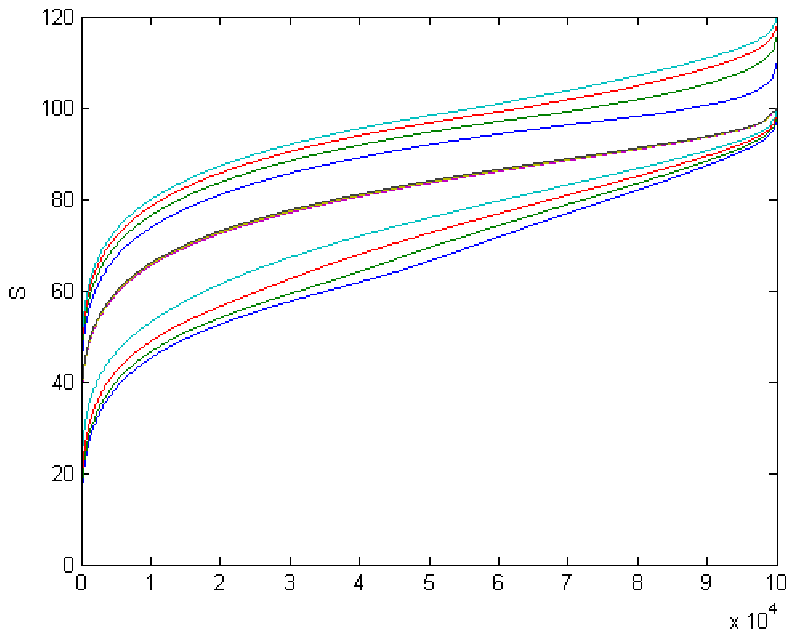

Figure 101: Sorted symmetrically distributed S.

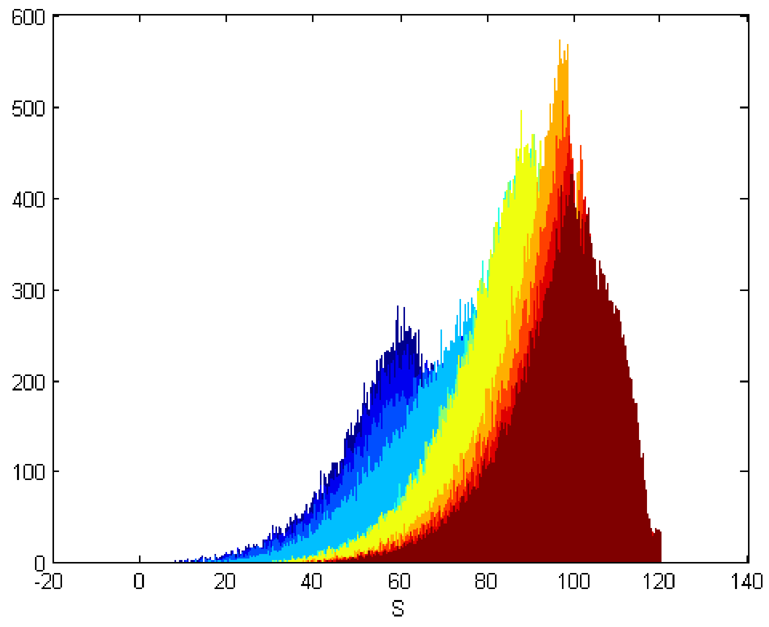

Figure 102: Histogram for symmetrically distributed S.

```
% Symmetricly Distributed S - price of underlying asset
S = zeros(n,nbrS);
firstDist=0.03;
% norminvMax=norminv(0.75);
% dist=(norminvMax-firstDist)*2/(nbrS-3);
dist=(2-firstDist)*2/(nbrS-3);
HAdj=H./exp(0.5826*sigma.*sqrt(dt));
% HAdj=90*ones(n,1);
for k=(-(nbrS-1)/2):((nbrS-1)/2)
    if k==-1 % S<HAdj and we cant get a S below HAdj-0.99*min(HAdj).
        S(:,k+((nbrS-
1)/2)+1)=max(HAdj.*(1+k*firstDist*sigma.*sqrt(dt)),HAdj-0.99*min(HAdj));
    elseif k==0 % the case when S is Hadj
        S(:,k+((nbrS-1)/2)+1)=HAdj;
    elseif k==1 % S>HAdj and we cant get a S above HAdj+0.99*min(HAdj).
        S(:,k+((nbrS-
1)/2)+1)=min(HAdj.*(1+k*firstDist*sigma.*sqrt(dt)),HAdj+0.99*min(HAdj));
    elseif k>1 % S>HAdj and we cant get a S above HAdj+0.99*min(HAdj).
        S(:,k+((nbrS-
1)/2)+1)=min(HAdj.*(1+(k*dist+firstDist).*sigma.*sqrt(dt)),HAdj+0.99*min(HA
dj));
    else % S<HAdj and we cant get a S below HAdj-0.99*min(HAdj).
        S(:,k+((nbrS-1)/2)+1)=max(HAdj.*(1+(k*dist-
firstDist).*sigma.*sqrt(dt)),HAdj-0.99*min(HAdj));
    end
end
```

8.2.2.7.2 Normally distributed S

We have also used a normal distribution for S with mean H_A. S is with a probability of 98 percent within $maxNbrStds$ (which we have assigned the value 1.6) standard deviations (of the log returns r_i) from H_A. We adjusted the distribution not to get negative values, still making sure that the distribution remained symmetric. What we practically do is to use *min* and *max* functions to make sure that no values of S are below one percent of the minimum H_A or above H_A plus 99 percent of the minimum H_A. This approach is very similar to the symmetric approach, the difference being that we now use a normal distribution, $k = \frac{1.6}{\phi^{-1}(0.99)} \cdot Y$ (where $Y \sim N(0,1)$), rather than fixed numbers of these standard deviations.

This distribution does not meet the first goal in a strict sense, but at least the *distribution* of S is symmetric around the adjusted barrier. It performs at least as good as the symmetric distribution with respect to all the other goals. It even performs substantially better when it comes to the sixth goal.

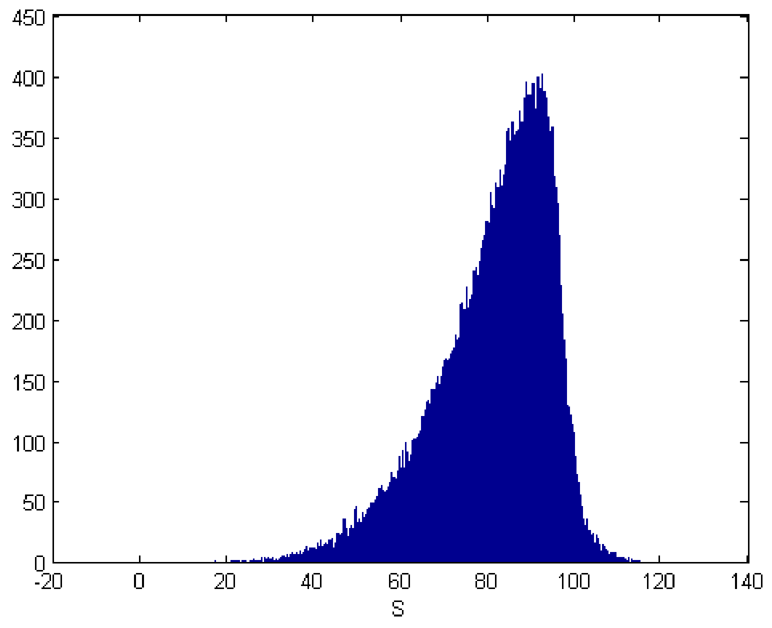

Figure 103: Histogram for normally distributed S.

```
% Normal Distrbuted(k) S - price of underlying asset
S = zeros(n,nbrS);
maxNbrStds = 1.6;
HAdj=H./exp(0.5826*sigma.*sqrt(dt));
for i=1:nbrS
    for j=1:n
        haltonS = uniform(j,nbrParaExS+i);
        HAdjTemp=HAdj(j);
        if haltonS<=0.5
            S(j,i) =
max(HAdjTemp*(1+maxNbrStds/norminv(0.99)*norminv(haltonS)*sigma(j)*sqrt(dt(
j))),HAdjTemp-0.99*min(HAdj));
            % Never negative value, normal distributed around HAdj and
"98%" within
            % maxNbrStds times the Std from HAdj.
        else
            S(j,i) =
min(HAdjTemp*(1+maxNbrStds/norminv(0.99)*norminv(haltonS)*sigma(j)*sqrt(dt(
j))),HAdjTemp+0.99*min(HAdj));
            % This will make it symmetrically distributed
        end
    end
end
```

8.2.3 Parameter simulation for problem 2

We now need to adjust some of the parameters that we use in problem 1 since we introduce a stub, $\Delta t^{first} < \Delta t$ (Δt is the same as in problem 1, see Footnote 61). The parameters that need to be changed are T,n_{monit}^{eq} and S.

8.2.3.1 Stub to first monitoring instant, Δt₁

The time to the first monitoring, Δt^{first}, should be less than Δt. We make the Δt^{first} uniformly distributed, $u\left(\frac{0.5}{365}, \Delta t\right)$.

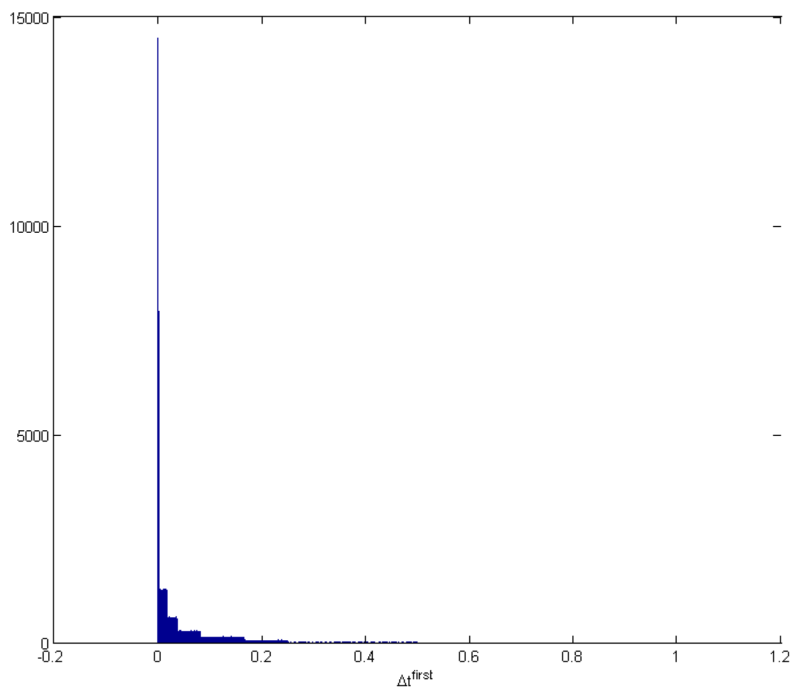

Figure 104: Histogram for Δt^{first}.

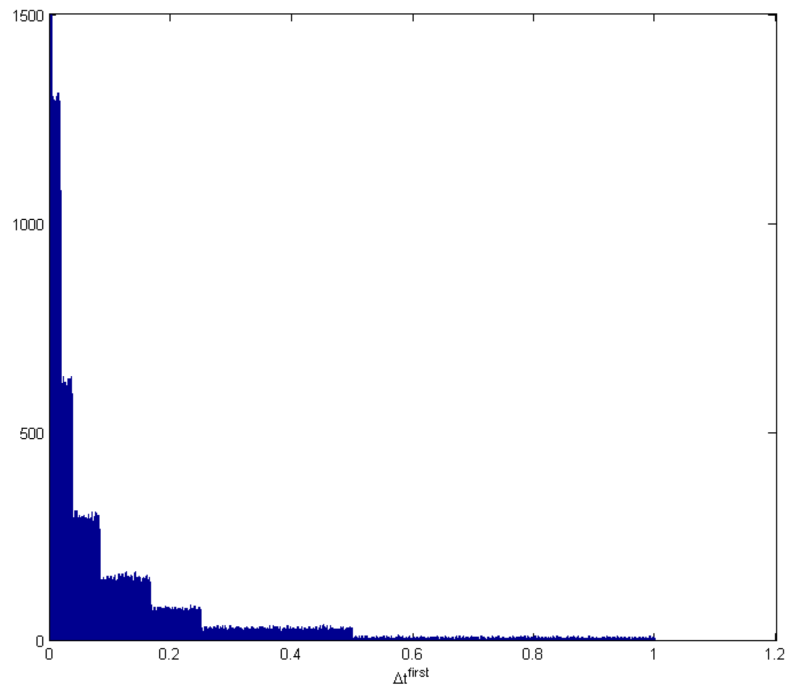

Figure 105: Zoomed in version of Figure 104.

```
%% t_first - time step to first monitoring date
halton_t_first = uniform(:, 7);
t_first = (dt-0.5/365).*halton_t_first + 0.5/365; %Min is a half day
```

8.2.3.2 Time to maturity, T, and number of monitoring instants, n_{monit}

There is now a new requirement that the parameters must satisfy,

$$T = (n_{monit} - 1)\Delta t + \Delta t^{first}.$$

Below is the code for the final n_{monit} and T. We need to be sure that there are at least two monitoring points, one for the stub and at least one time between monitoring instants of length Δt (problem 2 reduces to problem 1 when n_{monit} is equal to one). It should be pointed out that T can now take larger values than 11 since we have added two monitoring instants to all parameter sets to ensure that $n_{monit} \geq 2$.

```
%% The Final T - time to maturity
nMonit=zeros(n, 1);
for i = 1:n
    nMonit(i)=round((Ttemp(i)-t_first(i))/dt(i));
    nMonit(i)  = nMonit(i)  + 2;% We need nMonit to be at least 2.
    T(i)=(nMonit(i)-1)*dt(i)+t_first(i);
end
```

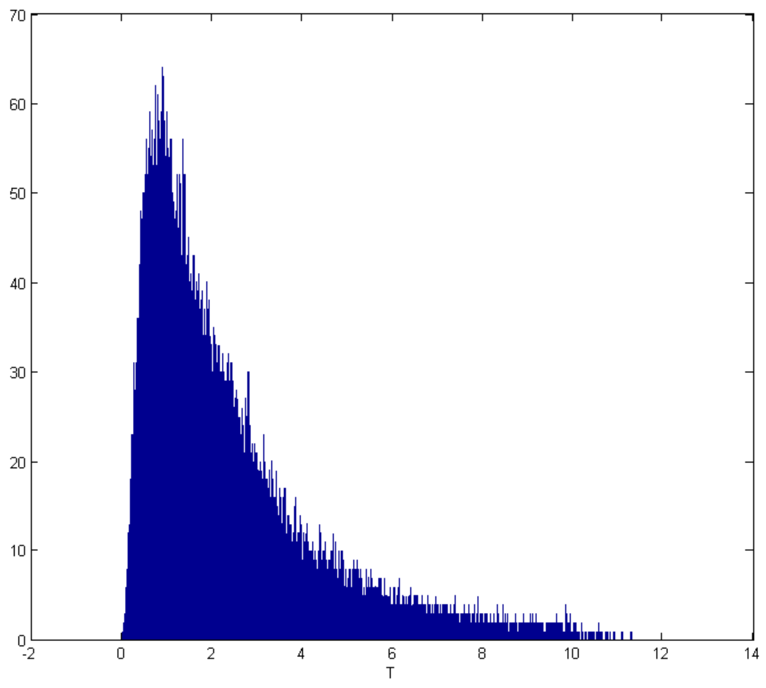

Figure 106: Histogram for T.

8.2.3.3 Underlying asset, S

8.2.3.3.1 Symmetrically distributed S

We now want S symmetric around H (instead of H_A) since that is where our weight factor changes the most. We use the same code as for problem one, but with three changes. Δt is replaced by Δt^{first}, H_A is replaced by H, and the width of the distribution is adjusted to make sure that the weight factor goes from approximately zero to approximately one[63]. The k's (number of standard deviations of the log returns R_i, with $\Delta t = \Delta t^{first}$) that we use to fulfill this (determined iteratively) are $k = \{-4.395, -1.815, -0.855, -0.435, -0.05, 0, 0.05, 0.435, 0.855, 1.815, 4.395\}$.

[63] We verify this for $\alpha = 3$.

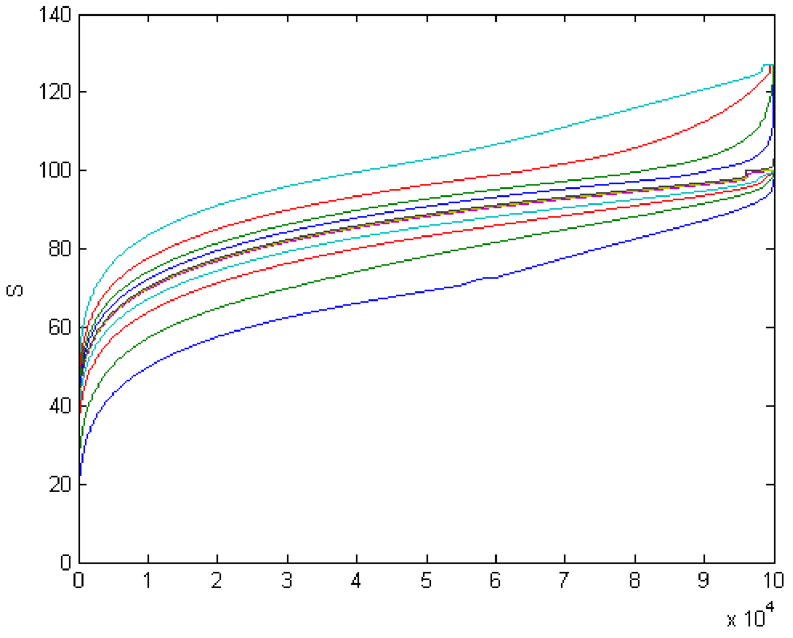

Figure 107: Sorted symmetrically distributed S.

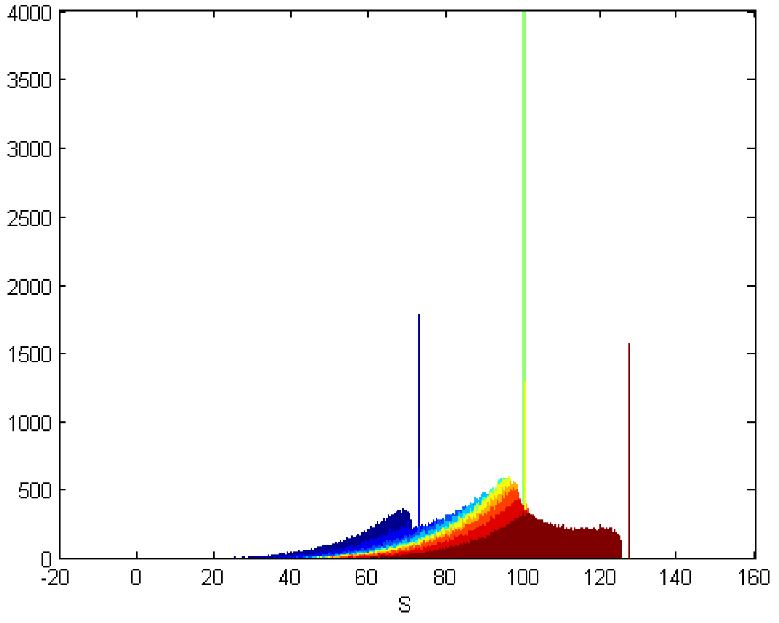

Figure 108: Histogram for symmetrically distributed S.

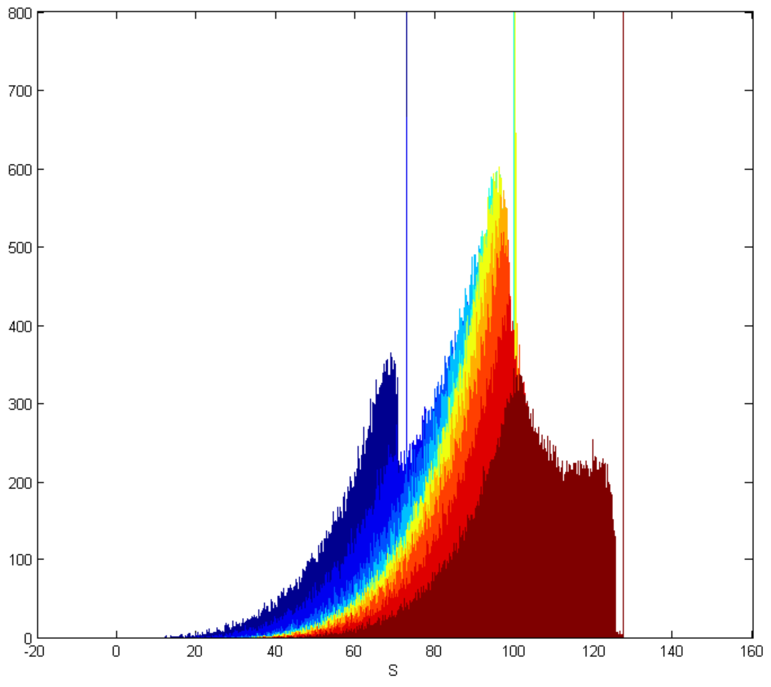

Figure 109: Zoomed in version of Figure 108.

```
% Symmetricly Distributed S around H with sqrt(t_first) - price of
underlying asset
S = zeros(n,nbrS);
firstDist=0.05;
secondDist=0.15;
dist=0.045;
% norminvMax=norminv(0.75);
% firstDist=0.03;
% dist=(norminvMax-firstDist)*2/(nbrS-3);
% H = ones(n,1)*90;
for k=(-(nbrS-1)/2):((nbrS-1)/2)
    if k==-1 % S<HAdj and we cant get a S below HAdj-0.99*min(HAdj).
        S(:,k+((nbrS-
1)/2)+1)=max(H.*(1+k*firstDist*sigma.*sqrt(t_first)),H-0.99*min(H));
    elseif k==0 % the case when S is Hadj
        S(:,k+((nbrS-1)/2)+1)=H;
    elseif k==1 % S>HAdj and we cant get a S above HAdj+0.99*min(HAdj).
        S(:,k+((nbrS-
1)/2)+1)=min(H.*(1+k*firstDist*sigma.*sqrt(t_first)),H+0.99*min(H));
    elseif k>1 % S>HAdj and we cant get a S above HAdj+0.99*min(HAdj).
        S(:,k+((nbrS-1)/2)+1)=min(H.*(1+(dist*(3^(abs(k)-
1))+secondDist*abs(k)).*sigma.*sqrt(t_first)),H+0.99*min(H));
    else % S<HAdj and we cant get a S below HAdj-0.99*min(HAdj).
        S(:,k+((nbrS-1)/2)+1)=max(H.*(1-(dist*(3^(abs(k)-
1))+secondDist*abs(k)).*sigma.*sqrt(t_first)),H-0.99*min(H));
    end
end
```

8.2.3.3.2 Normally distributed S

We also use a normal distribution for S. The simulated values of S are with a probability of 98 percent within $maxNbrStds$ (which we have assigned the value eight) standard deviations (of the log returns $Ꝝ_i$, with $\Delta t = \Delta t^{first}$) from H. $maxNbrStds$ is iteratively determined so as to make sure that the weight factor goes from approximately zero to approximately one over the interval[64]. The only other difference from the case with symmetrically distributed values of S is that k is now normally distributed, $k = \frac{8}{\phi^{-1}(0.99)} \cdot Y$ (where $Y \sim N(0,1)$).

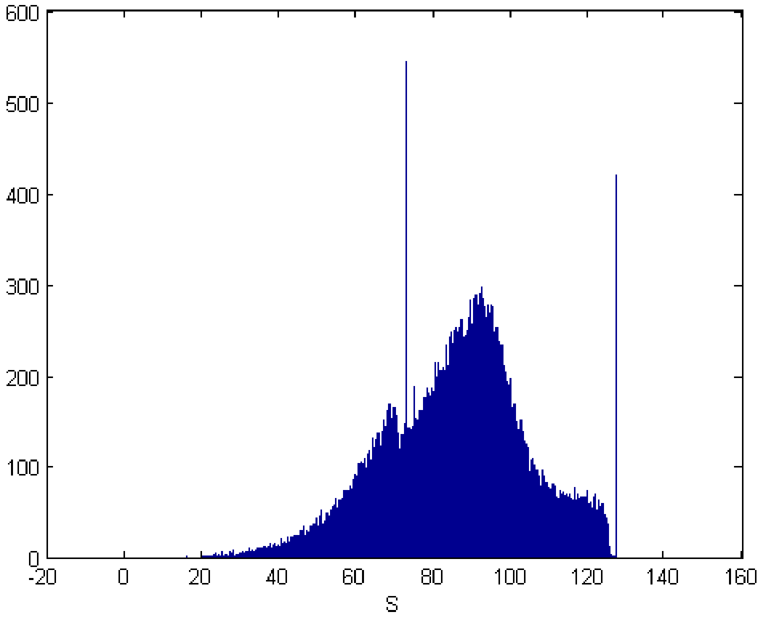

Figure 110: Histogram for normally distributed S.

```
% Normal Distrbuted(k) S - price of underlying asset
S = zeros(n,nbrS);
maxNbrStds = 8;
for i=1:nbrS
    for j=1:n
        haltonS = uniform(j,nbrParaExS+i);
        if haltonS<=0.5

S(j,i)=max(H(j)*(1+maxNbrStds/norminv(0.99).*norminv(haltonS)*sigma(j)*sqrt
(t_first(j))),H(j)-0.99*min(H));
            % Never negative value, normal distributed around HAdj and
"98%" within
            % maxNbrStds times the Std from H.
        else

S(j,i)=min(H(j)*(1+maxNbrStds/norminv(0.99).*norminv(haltonS)*sigma(j)*sqrt
(t_first(j))),H(j)+0.99*min(H));
            % This will make it symmetrically distributed
        end
    end
end
```

[64] We verify this for $\alpha = 3$.

8.3 Appendix C: Comments on the work process

In this section we briefly mention or describe activities that have been particularly time consuming in the work process.

Since the functions provided by SunGard were written in the C programming language, which we had no previous experience of, and since we therefore preferred to work in Matlab, a lot of time during the first weeks was devoted to

1. understanding C
2. getting C and Matlab to "work together"

A large part of our work has been made up of calculating option values corresponding to the simulated parameter values and using the loop approach to calculate values of α. This has been a bottleneck since these operations are fairly time consuming, even on modern computers. Furthermore, these processes have for the following reasons had to be repeated over and over again:

- alterations to the simulation approach suggested by our supervisors
- the fact that some parameters should be used as input for calculations in certain cases but not in others
- (programming) errors on our part

As for the different contributions of the two authors, we have collaborated in the true sense of the word. We have been sitting together each day (with only a handful of exceptions), working on computers next to each other and discussing more or less everything. It is thus not possible to point out for instance different sections of the thesis that are contributions from one author or the other.

Master's Theses in Mathematical Sciences 2014:E29
ISSN 1404-6342

LUTFMA-3261-2014

Mathematics
Centre for Mathematical Sciences
Lund University
Box 118, SE-221 00 Lund, Sweden

http://www.maths.lth.se/