May 2020

Analysis of Inventory Models with Random Supply Using a Long-Term Average Criterion

Lars Moestue
University of Wisconsin-Milwaukee

ANALYSIS OF INVENTORY MODELS WITH RANDOM SUPPLY USING A LONG-TERM AVERAGE CRITERION

by

Lars Moestue

A Thesis Submitted in

Partial Fulfillment of the

Requirements for the Degree of

Master of Science

in Mathematics

at

The University of Wisconsin-Milwaukee

May 2020

ABSTRACT

ANALYSIS OF INVENTORY MODELS WITH RANDOM SUPPLY USING A LONG-TERM AVERAGE CRITERION

by

Lars Moestue

The University of Wisconsin-Milwaukee, 2020
Under the Supervision of Professor Richard Stockbridge

In this thesis we will use different numerical algorithms for inventory models, where the inventory level is described by a stochastic differential equation and therefore random. Furthermore we assume that order supply is randomly distributed. The goal is to find the optimal order strategy to minimize the long-term average costs.

This stochastic problem can be reformulated as non-linear optimization problem. However the problems are too complex to solve by hand, so we need to use numerical optimization algorithms and for some of the models even numerical integration methods.

These algorithms then can be used to analyze some properties and make sensitivity analyses of different model parameters for inventory models based on a Brownian motion with different distributions for the supply. In this thesis we will see that this method works for a wide range of supply distributions. Some of them are relatively easy like the uniform distributions. Others are discrete like the binomial distribution. But there are also more complex distributions like the beta distribution and the truncated normal distribution, where this approach works as well.

Dedicated to my parents who always support my academic and personal endeavor
and to Lena whose love strengthens me every day.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

First and foremost, I would like to thank Prof. Richard Stockbridge for giving me the opportunity to work on this interesting project and for and all the fruitful discussions we had together.

Secondly, I would like to thank Prof. Zhu and Prof. Spade for being part of my thesis committee.

Lastly, I would like to thank Prof. Boyd for the great support in these crazy and difficult times. In situations like this it is important to have a leader like her, who helps everybody to navigate through the difficulties everybody faces now.

1. Introduction

One of the most important fields for applied mathematics nowadays are the numerical mathematics. Over the last 70 years this field made an enormous progress to solve complex problems, which seem to be unsolvable by hand. Especially the numerical optimization plays an important role here. The advanced algorithms from this field have wide applications for different problems in mathematics and beyond the borders of mathematics in engineering, physics and chemistry. The most common algorithm for constrained non-linear optimization problems is the Sequential Quadratic Programming (SQP). This algorithm was developed in the 1960s and 1970s.

The focus of this thesis is to use the SQP algorithm to find the optimal ordering points in inventory models. The inventory process is described by Stochastic Differential Equation. Our focus will be on such models with a Brownian motion. However, not only the the inventory model has a random behavior, but also the proportion of the supply that is delivered will be random. These models lead to complex cost structures that can not be optimized by hand (a detailed discussion about this can be found in Chapter 2). This makes it very difficult to make general statements about the influence of different model parameters or different supply distributions. But with the help of numerical algorithms such as the SQP-algorithm and the Gauss-Kronrod-quadrature for numerical integration, the optimal ordering points can be found in a short time. The algorithms used are described in the appendix.

As we will see in Chapter 3, this approach works not only for a classical Brownian motion inventory model, but also for models with reflected Brownian motions and geometric Brow-

nian motion. Furthermore, this method also works for a very wide range of possible supply distributions. In some cases as for uniform distributions the cost function will be relatively simple. However, for most distributions such as beta distribution, the truncated normal distribution, or polynomials distributions, the cost function contains integrals, which do not have (simple) antiderivatives. Still others, especially discrete distributions, such as the binomial distributions have sums, partly with many terms in the cost function. Nevertheless, for all of the above distributions, this numerical approach was able to solve the optimization problems in a short time. All of the algorithms we use in this thesis have running times below five minutes, although to make general statements about the model at least 500 data points were used each time.

2. Inventory models and optimal solutions

In this chapter we give a quick overview over the mathematical theory behind the models. We only summarize the key results that are important for this thesis. For a more detailed discussion see, [2], [3] and [4]. The inventory process is modeled as solution of the stochastic differential equation:

$$\mathrm{d}X_0(t) = \hat{\mu}(X_0(t))\mathrm{d}t + \hat{\sigma}(X_0(t))\mathrm{d}W(t), \quad X_0(0) = x_0, \tag{2.1}$$

with $\hat{\mu} : \mathbb{R} \to \mathbb{R},\ \hat{\sigma} : \mathbb{R} \to \mathbb{R}^+$, which are taking values in an interval $\mathcal{I} = (a, b)$. We assume the following condition to hold.

Condition 2.1 *i) The scale function*

$$S(x) = \int^x s(y)\mathrm{d}y$$

and the speed measure

$$M(y, z) := \int_y^z \frac{1}{\hat{\sigma}^2(x)s(x)}\mathrm{d}x$$

with

$$s(x) = \exp\left(-\int^{v} \frac{2\hat{\mu(y)}}{\hat{\sigma}^2(y)} \mathrm{d}v \right)$$

are absolutely continuous with respect to the Lebesgue-measure.

ii) The left boundary a is attracting, while the boundary b is non-attracting. If b is a natural boundary, $M(y,b) < \infty \ \forall y \in (a,b)$. If $a = -\infty$ or $b = \infty$ then the corresponding boundary is natural.

The state space of the process X is denoted by $\mathcal{E}$ (includes a only if its attainable and b only if its an entrance) and $\bar{\mathcal{E}}$ is the closure of $\mathcal{E}$ in $\mathbb{R}$. Let $\mathcal{R} = \{(y,z) \in \mathcal{E}^2) : y < z\}$, where y denotes the pre-order inventory level and z the nominal post-ordered inventory level and $\bar{\mathcal{R}} = \{(y,z) \in \bar{\mathcal{E}}^2 : y \leq z\}$. The actual post-order inventory is given as a realization of a kernel distribution $\mathcal{Q}(\cdot, y, z)$, which depends on y and z. The set of all kernel distributions is defined as $\mathfrak{Q} = \{\mathcal{Q}(\cdot, y, z) : (y,z) \in \bar{\mathcal{R}}\}$. We assume the following conditions to be true:

Condition 2.2 *i) $\forall y \in \mathcal{E}, \ \mathcal{Q}(\cdot, y, y) = \delta_y(\cdot)$, which is the Dirac measure on $\{y\}$.*

ii) $\forall (y,z) \in \mathcal{R}$:

a) supp$(\mathcal{Q}(\cdot, y, z)) \subset [y, z]$ and $\mathcal{Q}(\{y\}, y, z) = 0$;

b) for all sequences $(y_n, z_n)_{n\in\mathbb{N}} \subset \mathcal{R}$, such that $y_n \to y$ and $z_n \to z$ for $n \to \infty$, it holds that $\mathcal{Q}(\cdot, y_n, z_n) \xrightarrow{\mathcal{D}} \mathcal{Q}(\cdot, y, z)$.

An admissible nominal ordering policy (τ, Y) is a sequence $\{(\tau_k, Y_k)\}_{k\in\mathbb{N}}$ that fulfills

i) $(\tau_k)_{k\in\mathbb{N}}$ is a strictly increasing sequence of $\{F_{t-}\}$-stopping times;

ii) $\forall n \in \mathbb{N}, \ Y_n \in \mathcal{E}$ is $\{F_{t-}\}$-measurable, such that $Y_n > X(\tau-)$;

iii) the average cost $J(t, Y)$ of the this ordering policy given by (2.2) (see below) is finite.

τ_k is the time at which the kth order is placed (increasing, since it is not possible to place the $(k+1)$st order before the kth order) and Y_k is the (nominal) size of the order k. This leads to the following stochastic process X for the inventory level:

$$X(t) = x_0 + \int_0^t \mu(X(s))\mathrm{d}s + \int_0^t \sigma(X(s))\mathrm{d}W(s) + \sum_{k=1}^{\infty} \mathbb{1}_{(\tau_k \le t)} \tilde{Y}_k,$$

with $\tilde{Y}_k$ being randomly distributed on $(0, Y_k)$. It is possible that the inventory level at time $0-$ is such that an order is placed at time $t = 0$. Furthermore, observe that $X(\tau_k-)$ is the inventory level just before order k and we assume that order is delivered instantly, so $X(\tau_k)$ is the inventory level with the new inventory.

Let $c_0 : (a, b) \to \mathbb{R}^+$ be the holding/back-order cost rate and $c_1 : \bar{\mathcal{R}} \to \mathbb{R}^+$ be the ordering cost function, which fulfill the following Condition.

Condition 2.3 *i)* c_0 *is continuous and both limits*

$$c_0(a) := \lim_{x \to a} c_0(x) \text{ and } c_0(b) := \lim_{x \to b} c_0(x)$$

exist in $\bar{\mathbb{R}}^+$*, with* $c_0(\pm\infty) = \infty$*. Furthermore we require*

$$\int_y^b c_0(v)\mathrm{d}M(v) < \infty \ \forall y \in \mathcal{I}.$$

ii) c_1 *is continuous and there exists a constant* $k_1 > 0$ *such that* $c_1 \ge k_1$*. Moreover the function*

$$\widehat{c}_1(y, z) := \int_y^z c_1(y, v)\mathcal{Q}(dv, y, z)$$

is continuous.

For the ordering policy (τ, Y) the long-term average expected holding/back-order cost plus

ordering cost is defined as:

$$J_0(\tau, Y) := \limsup_{t\to\infty} \frac{1}{t}\mathbb{E}\left(\int_0^t c_0(X(s))\mathrm{d}s + \sum_{k=1}^{\infty} \mathbb{1}_{\{\tau_k\le t\}} c_1(X(\tau_k-), \tilde{Y}_k)\right) \tag{2.2}$$

The goal is to find the ordering policy (τ^*, Y^*), such that

$$J_0(\tau^*, Y^*) = \min\{J_0(\tau, Y) : (\tau, Y) \text{ admissible nominal ordering policy}\}. \tag{2.3}$$

For this purpose we need to define two auxiliary functions.

Definition 2.4

Let $x_0 \in \mathcal{E}$ be the initial position of the process X_0. Then the functions g_0, ζ: $\mathcal{I} \to \mathbb{R}$ are defined as

$$\begin{aligned} \zeta(x) &:= \int_{x_0}^{x}\int_{u}^{b} 2\mathrm{d}M(v)\mathrm{d}S(u) \\ \textit{and } g_0(x) &:= \int_{x_0}^{x}\int_{u}^{b} 2c_0(v)\mathrm{d}M(v)\mathrm{d}S(u). \end{aligned}$$

In case of an inventory process (2.1) *the functions g_0 and ζ are solutions of the differential equations*

$$Af = -c_0,\ f(x_0) = 0 \qquad \textit{and} \qquad Af = -1,\ f(x_0) = 0,$$

with $Af = \mu f' + \frac{\sigma^2 f''}{2}$

Both functions can be extended to the boundaries by continuity, where they may take the value $\pm\infty$.

The class that optimizes (2.3), are the (s, S)-policies (cf. [4]).

Definition 2.5

Let $(y, z) \in \mathcal{R}$ the nominal (y, z)-ordering policy (τ, Z) is defined such that $\tau_0 = 0,\ \tau_1 =$

$\inf\{t \geq \tau_0 : X(t-) \leq y\}$ *and*

$$\tau_k = \inf\{t > \tau_{k-1} : X(t-) \leq y\} \; \forall k \geq 2, \;\; Z_k = z, \;\; \forall k \in \mathbb{N}$$

For a function $f : \mathcal{I} \to \mathbb{R}$ and a kernel distribution $\mathcal{Q}(\cdot, y, z)$ we define

$$\widehat{B}_{\mathcal{Q}} f(y, z) \int_y^z f(v)\mathcal{Q}(\mathrm{d}v, y, z) - f(y).$$

With this notation we can define one of the main results for the analysis.

Theorem 2.6

Let $(y, z) \in \mathcal{R}$. Then the long-term average cost of the corresponding (y, z)-nominal ordering policy (τ, Z) under the kernel distribution $\mathcal{Q}(\cdot, y, z)$ is given by

$$J_0(\tau, Z) = \frac{\widehat{c}_1(y, z) + \widehat{B}_Q g_0(y, z)}{\widehat{B}_{\mathcal{Q}} \zeta(y, z)}. \tag{2.4}$$

3. Models with drifted Brownian motion

In this chapter we will use the algorithms and formulas from the last chapter to compute the optimal (y, z)-policies and the corresponding costs for some models, which includes a Brownian motion an>d a random supply with distribution $\mathcal{Q}(\cdot, y, z)$. For all the models Condition 2.1 and Condition 2.3 i) are fulfilled, see [2] and [3] . If furthermore $\mathcal{Q}(\cdot, y, z)$ is an absolutely continuous probability measure Condition 2.3 ii) is fulfilled, too (as $\widehat{c_1}$ then is an integral of a continuous function). All graphs in this chapter use at least 500 data points per line, which is, in the case of an interval length of 1, a step size of around 0.002 for curves with peaks or steep increases/decreases 5000 data points were used, to make sure that there are no violation of the continuity assumption.

3.1. Classical model

Let

$$\mathrm{d}X_0(t) = -\mu \mathrm{d}t + \sigma \mathrm{d}W(t),\ X_0(0) = x_0 \in \mathcal{I} := (-\infty, \infty) \tag{3.1}$$

with $\mu, \sigma > 0$ and W a standard Brownian motion. This means the model allows negative as well as positive inventory levels. We define the cost functions $c_0 : \mathbb{R} \to \mathbb{R}^+$ and $c_1 : \bar{\mathcal{R}} :=$

$\{(y,z) \in \mathbb{R}^2 : y \le z\} \to \mathbb{R}^+$ as

$$c_0(x) = \begin{cases} -c_b x & x \le 0 \\ \\ c_h x & x > 0 \end{cases}, \qquad c_1(y,z) = k_1 + k_2(z-y),$$

where $c_b, c_h, k_1, k_2 > 0$. In this model we do not assume any economies of scale, so there is no incentive to make larger orders. Furthermore, back-order cost rate and holding cost rate can be different.

Theorem 3.1

Let

$$\hat{g}_0(x) = \begin{cases} -\frac{c_b}{2\mu}x^2 - \frac{\sigma^2 c_b}{2\mu^2}x + \frac{\sigma^4(c_b+c_h)}{4\mu^3}\left(\exp\left(\frac{2\mu}{\sigma^2}x\right) - 1\right) & x \le 0 \\ \frac{c_h}{2\mu}x^2 + \frac{\sigma^2 c_h}{2\mu^2}x & x > 0. \end{cases}$$

Then in the model stated above, the functions g_0 and ζ are given by

$$g_0(x) = \hat{g}_0(x) - \hat{g}_0(x_0) \text{ and } \zeta(x) = \frac{x - x_0}{\mu}, \quad x \in \mathbb{R}.$$

Proof. We calculate

$$-\mu g_0'(x) = \begin{cases} c_b x + \frac{\sigma^2 c_b}{2\mu} - \frac{\sigma^2(c_b+c_h)}{2\mu}\exp\left(\frac{2\mu}{\sigma^2}x\right) & x \le 0 \\ -c_h x - \frac{\sigma^2 c_h}{2\mu} & x > 0 \end{cases}$$

and

$$\frac{\sigma^2}{2}g_0''(x) = \begin{cases} -\frac{\sigma^2 c_b}{2\mu} + \frac{\sigma^2(c_b+c_h)}{2\mu}\exp\left(\frac{2\mu}{\sigma^2}x\right) & x \le 0 \\ \frac{\sigma^2 c_h}{2\mu} & x > 0, \end{cases}$$

so $Ag_0 = -c_0,\ g_0(x_0) = 0$. Similar we can compute $A\zeta = -1,\ \zeta(x_0) = 0$. $\square$

3.1.1. Uniform distribution

We assume that the supply of a nominal (y, z) order is uniformly distributed on $\mathcal{L}^a = [p_a, z]$, where $p_a = (1-a)y + az$ and $a \in [0, 1]$. If a is close to one, there is only a small spread and an order is even in the worst-case scenario close to the nominal order. However, a small a leads to a much higher spread and thus to high uncertainty in the actual delivery of the order, since it is possible to get only a low fraction of the actual order. In this case the kernel distribution is given by

$$\mathcal{Q}(\mathrm{d}v, y, z) = \frac{\mathbb{1}_{\mathcal{L}^a}(v)\mathrm{d}v}{z - p_a} = \frac{\mathbb{1}_{\mathcal{L}^a}(v)\mathrm{d}v}{(1-a)(z-y)} = \frac{\mathbb{1}_{\mathcal{L}^a}(v)\mathrm{d}v}{q_a} \tag{3.2}$$

Theorem 3.2

The distribution $\mathcal{Q}(\mathrm{d}v, y, z)$ *from* (3.2) *fulfills Condition* 2.2.

Proof. i) a uniform distribution on $[y, y]$ is a deterministic distribution on $\{y\}$

ii) a) $\mathrm{supp}(\mathcal{Q}(\cdot, y, z)) = [p_a, z] \subset [y, z]$ and $\mathcal{Q}(\{y\}, y, z) = 0$, since $\mathcal{Q}(\cdot, y, z)$ is absolutely continuous.

b) it is easy to check, that $U([(1-a)y_n + az_n, z_n]) \xrightarrow{\mathrm{d}} U([(1-a)y + az, z])$ for $y_n \to y$ and $z_n \to z$.

□

We now compute

$$\widehat{c}_1(y, z) = \int_{p_a}^{z} \frac{k_1 + k_2(v-y)}{q_a}\mathrm{d}v = k_1 + k_2\left(\frac{z^2 - p_a^2}{2q_a} - y\right) = \frac{(1+a)z + (1-a)y}{2} \text{ and}$$
$$\widehat{\zeta}(y, z) := \frac{1}{q_a}\int_{y}^{z} \zeta(v)\mathcal{Q}(\mathrm{d}v, y, z) = \int_{p_a}^{z} \frac{\zeta(v)}{q_a}\mathrm{d}v = \frac{1}{q_a\mu}\left(z^2 - p_a^2\right).$$

For $\widehat{g}_0(y, z)$ we have to distinguish between three cases. The first case is $p_a > 0$, which leads

to

$$
\begin{aligned}
\widehat{g}_0(y,z) &:= \int_y^z g_0(v)\mathcal{Q}(\mathrm{d}v,y,z) \\
&= \int_{p_a}^z \left(\frac{c_h}{2\mu}v^2 + \frac{\sigma^2 c_h}{2\mu^2}v\right)\frac{\mathrm{d}v}{q_a} = \frac{1}{q_a}\left(\frac{c_h}{6\mu}\left(z^3 - p_a^3\right) + \frac{\sigma^2 c_h}{4\mu^2}\left(z^2 - p_a^2\right)\right)
\end{aligned}
$$

For the second case we assume $z \leq 0$:

$$
\begin{aligned}
\widehat{g}_0(y,z) &= \int_{p_a}^z \left(-\frac{c_b}{2\mu}v^2 - \frac{\sigma^2 c_b}{2\mu^2}v + \frac{\sigma^4(c_b+c_h)}{4\mu^3}\left(\exp\left(\frac{2\mu}{\sigma^2}v\right) - 1\right)\right)\frac{\mathrm{d}v}{q_a} \\
&= -\frac{1}{q_a}\left(\frac{c_b}{6\mu}\left(z^3 - p_a^3\right) + \frac{\sigma^2 c_b}{4\mu^2}\left(z^2 - p_a^2\right)\right) \\
&+ \frac{\sigma^4(c_b+c_h)}{4\mu^3}\left(\frac{\sigma^2}{2\mu q_a}\left(\exp\left(\frac{2\mu}{\sigma^2}z\right) - \exp\left(\frac{2\mu}{\sigma^2}p_a\right)\right) - 1\right)
\end{aligned}
$$

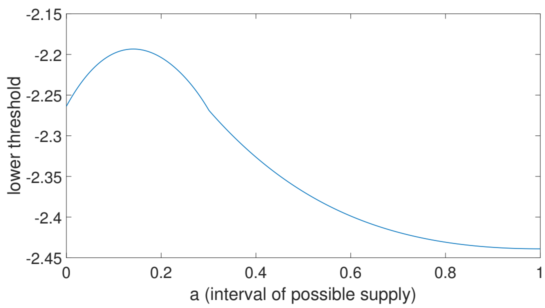
The last case is a mix of both cases before, where $z > 0$ and $(1-a)y + az \leq 0$:

$$
\begin{aligned}
\widehat{g}_0(y,z) &= \int_{p_a}^0 \left(-\frac{c_b}{2\mu}v^2 - \frac{\sigma^2 c_b}{2\mu^2}v + \frac{\sigma^4(c_b+c_h)}{4\mu^3}\left(\exp\left(\frac{2\mu}{\sigma^2}v\right) - 1\right)\right)\frac{\mathrm{d}v}{q_a} \\
&+ \int_0^z \left(\frac{c_h}{2\mu}v^2 + \frac{\sigma^2 c_h}{2\mu^2}v\right)\frac{\mathrm{d}v}{q_a} \\
&= \frac{1}{q_a}\left(\frac{1}{6\mu}\left(c_h z^3 + c_b p_a^3\right) + \frac{\sigma^2}{4\mu^2}\left(c_h z^2 + c_b p_a^2\right)\right) \\
&+ \frac{\sigma^4(c_b+c_h)}{4\mu^3 q_a}\left(\frac{\sigma^2}{2\mu}\left(1 - \exp\left(\frac{2\mu}{\sigma^2}p_a\right)\right) + p_a\right)
\end{aligned}
$$

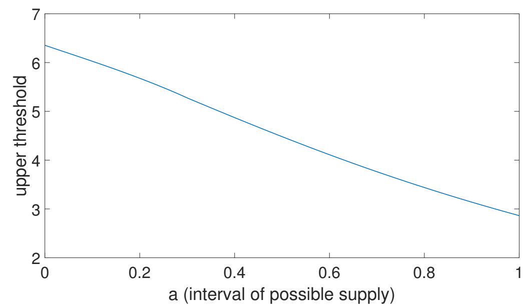
For the analysis we now set

$$
\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ c_h = 3,\ c_b = 4. \tag{3.3}
$$

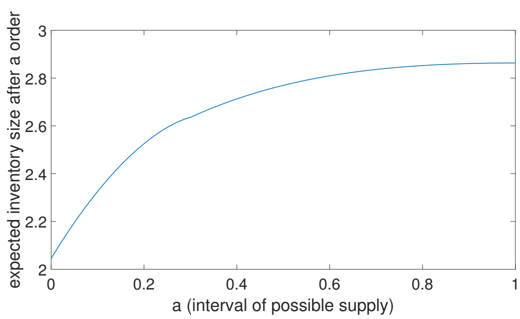
This means we have a large drift compared to the standard deviation. Furthermore we have high fixed costs for an order and the back-order costs are larger than the holding costs. As we can see in Figure 3.1(d) in the average cost is strictly decreasing with growing a. The average price for the deterministic model ($a = 1$) is around 15% lower than for the



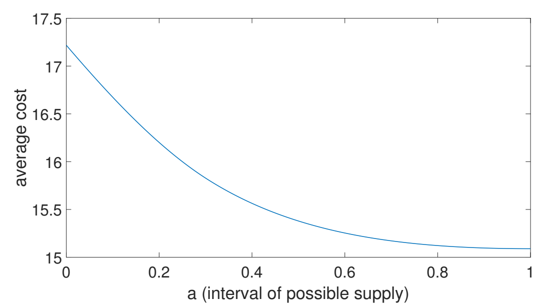
(a) lower threshold



(b) upper threshold



(c) expected inventory level after supply

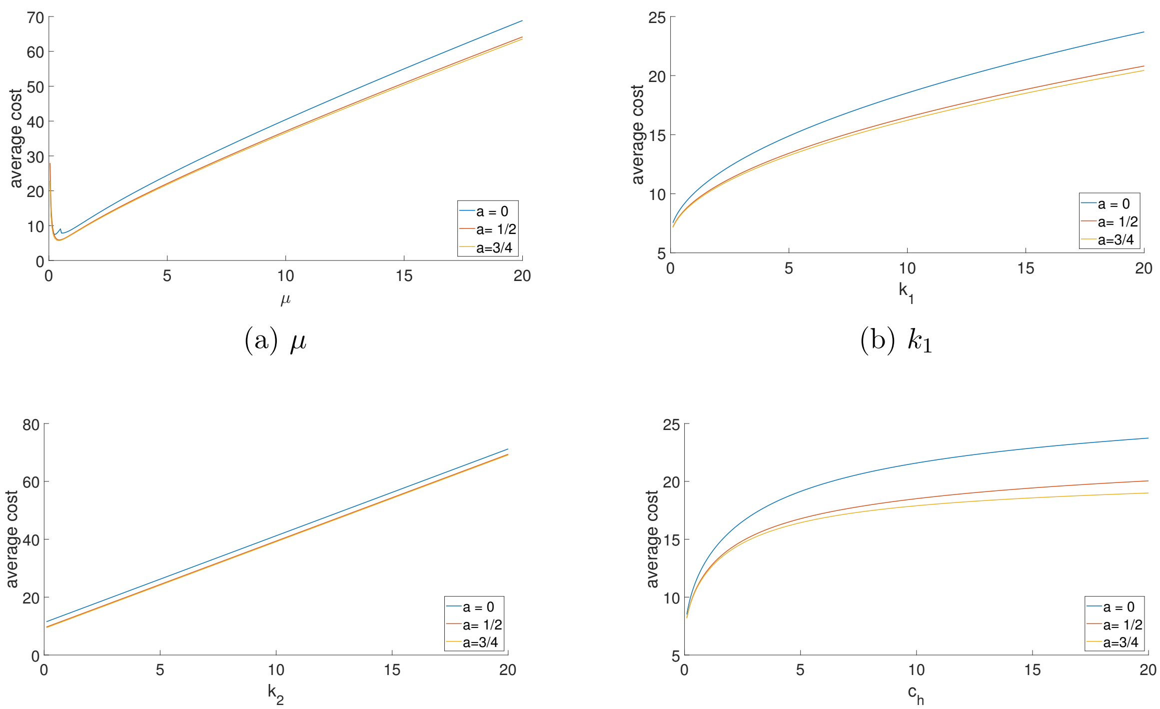


(d) average cost

Figure 3.1.: a-dependence in a drifted Brownian motion inventory model with uniformly distributed supply ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ c_h = 3,\ c_b = 4$)

random supply model with $a = 0$ (in this case the lowest supply is getting nothing). This is an intuitive result, since the expected inventory level after supply (cf. Figure 3.1(c)), given by $\frac{p_a+z}{2} = \frac{(1+a)z+(1-a)y}{2}$, is increasing in a to prevent high holding costs, which could arise in the case of higher deliveries than the expected value. However, by decreasing the expected supply, more orders are necessary and therefore the fixed order costs k_1 is paid more frequently. On the other hand, despite the fact that the expected inventory level after supply is increasing, the upper threshold (cf. Figure 3.1(b)) is (nearly linearly) decreasing and is more than halved for $a = 1$ in comparison to $a = 0$. So part of the higher average cost for small a is that there will be higher inventory levels after a supply and thus the holding costs are more expensive. The lower threshold (cf. Figure 3.1(b)) has a much lower spread. The difference between the maximum point of order and the minimum point of order is only around 0.25. First there is a small increase in the lower threshold until the maximum at around $a = 0.15$ and then a decrease afterwards. The reason that the absolute value of the upper threshold is larger than the the absolute value of the lower threshold is that $c_h < c_b$, which means it is cheaper to have a positive amount than to have negative amount.

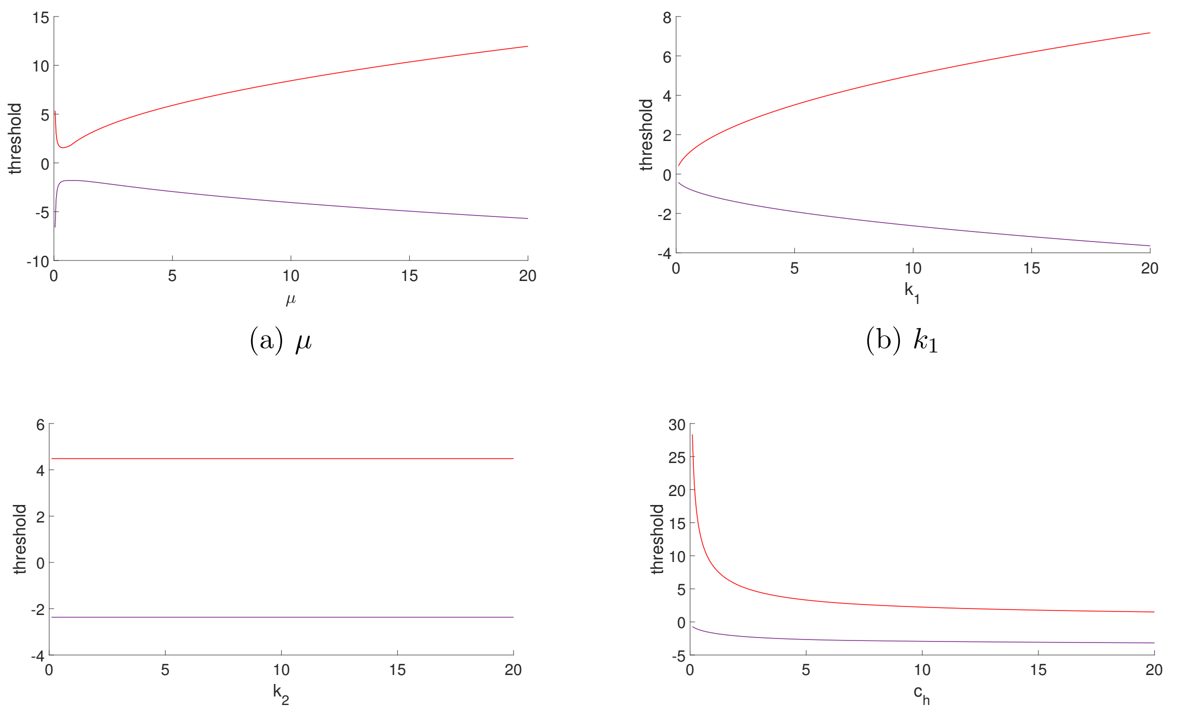
Our next goal is to perform a sensitivity analysis regarding some of the model parameters. We see in Figure 3.2 the average cost for three a $(a = 0, a = 0.5, a = 0.75)$ depending on four parameters (μ, k_1, k_2, c_h). All the other respective parameters are fixed as above in (3.3). Figure 3.3 shows the corresponding order thresholds for $a = \frac{1}{2}$. For all four parameters the cost is the highest for $a = 0$ and the lowest for $a = 0.75$, which can be explained with less uncertainty on the supply as above. For k_2 (cf. Figure 3.2(c)) the average cost is a linearly increasing function. The reason is that the threshold in this case stays constant and the cost function $J(\tau, Y)$ is a linear function in k_2. The thresholds (cf. Figure 3.3(c)) remain constant, because there are no economics of scale savings and only the actual supply needs to paid. For k_1 (cf. Figure 3.2(b)) and c_h (cf. Figure 3.2(d)) we have have an strong increase close to zero, which flattens out as k_1 and c_h increase. Both curves are very also similar with the values they take. For k_1 the upper and the lower thresholds (cf. Figure 3.3(b)) are



(a) μ (b) k_1

(c) k_2 (d) c_h

Figure 3.2.: Sensitivity analysis of the average cost in a drifted Brownian motion inventory model with uniformly distributed supply for different model parameters ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ c_h = 3,\ c_b = 4$, if not variable)



(a) μ

(b) k_1

(c) k_2

(d) c_h

Figure 3.3.: Sensitivity analysis of the thresholds in a drifted Brownian motion inventory model with uniformly distributed supply for different model parameters ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ c_h = 3,\ c_b = 4, a = \frac{1}{2}$, if not variable)

close to 0 for small k_1 since in this case it is very cheap to order. This makes it favorable to order very often and only maintain a small inventory. With an increasing k_1 the absolute value of both thresholds increases, which leads to fewer orders. The total costs increase, since fewer order become more expensive, while on the other hand the holding costs become more expensive as well, since the inventory level reaches more extreme values. For low c_h the upper threshold is comparably high, while the lower threshold (cf. Figure 3.3(d)) is close to zero, since in this case it is comparable very cheap to have positive inventory levels than negative inventory levels. However this advantage decreases with rate $\frac{1}{c_h}$, so the upper threshold decreases quickly for increasing c_h. Therefore the average price increases quickly near $c_h = 0$. At around $c_h = 6$, the absolute value of the lower threshold is smaller than the upper threshold. The reason is that this only happens at $c_h = 6$ and not at $c_h = c_b = 4$ is that the lower threshold is reached every time, while the upper threshold only is reached with a perfect delivery (which will almost surely not happen), so even with a greater upper threshold at $c_h = 4$ the expected actual supply is lower than $|c_b|$. For μ (cf. Figure 3.2(a)) we can see a more surprising result. The average cost is very high for μ close to zero and then decreases quickly until $\mu \approx 0.3$, where it has a minimum and starts growing linearly. The same holds for the thresholds (cf. Figure 3.3(a)), with absolute values both increasing very fast, until they move away again from each other. The reason for this is that in the case of very small μ the holding costs is very high, since there is only a small downward drift to the inventory. From a mathematical point of view for $\mu = 0$ the boundaries a and b would not be natural anymore and therefore Condition 2.1 ii) would be violated.

3.1.2. Binomial distribution

In this section we assume that the supply can only take finitely many values. The probability for $n + 1$ different possibilities is binomial distributed with probability p and parameter n.

The possible values of $\mathcal{Q}(\cdot,y,z)$ are $\left\{x_{k,y,z}=\frac{(n-k)y}{n+1}+\frac{(k+1)z}{n+1}:k\in\{0,\dots,n\}\right\}$, with

$$\mathcal{Q}\left(\left\{\frac{ky}{n+1}+\frac{(n+1-k)z}{n+1}\right\},y,z\right)=\binom{n}{k}p^k(1-p)^{n-k}=p_k.$$

With this definition there is no mass on y, which is a requirement of Condition 2.2. In this case the average cost function is given by

$$J(\tau,Y)=\frac{\sum_{k=0}^{n}\left(p_k(c_1(x_{k,y,z},z)+p_k g_0(x_{k,y,z})\right)-g_0(y)}{\sum_{k=0}^{n}p_k\zeta(x_{k,y,z})-\zeta(y)}$$

Theorem 3.3

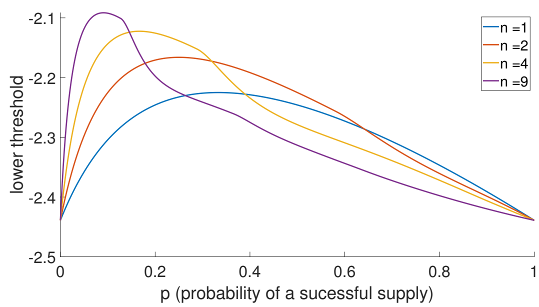
The binomial distribution as above fulfills Condition 2.3 ii) and Condition 2.2.

Proof. Condition 2.3 ii) is fulfilled, since c_1 is continuous and $q_{k,y_n,z_n}\to q_{k,y,z}$ $(n\to\infty)$. If $y=z$, $x_{k,y,z}=y$ $\forall k$ and therefore $\mathcal{Q}(\cdot,y,y)=\delta(\cdot)$. Obviously $x_{k,y,z}\in[y,z]$ $\forall k$, so $\operatorname{supp}(\mathcal{Q}(\cdot,y,z))\subset[y,z]$. So together with the fact that $y\neq x_{k,y,z}$ $\forall k$, which leads to $\mathcal{Q}(\{y\},y,z)=0$ we conclude that Condition 2.2 ii a) is fulfilled. Let now $(y_n,z_n)\subset\mathbb{R}$, such that $y_n\to y,\ z_n\to z$ for $n\to\infty$, furthermore we define F_n as distribution function of $\mathcal{Q}(\cdot,y_n,z_n)$ and F as distribution of $\mathcal{Q}(\cdot,y,z)$. Then since $q_{k,y_n,z_n}\to q_{k,y,z}$, we have

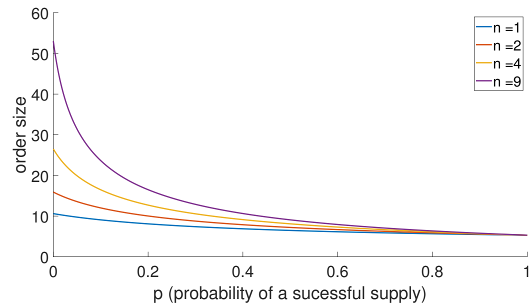
$$\lim_{n\to\infty}F_n(x)=F(x)\ \forall x\in[y,z]\setminus\{x_{k,y,z},\ k\in\{1,\dots,n\}\}.$$

Because F is not continuous in $x_{k,y,z}$, $\mathcal{Q}(\cdot,y_n,z_n)\xrightarrow{\mathrm{d}}\mathcal{Q}(\cdot,y,z)$, iib) of Condition 2.2 is fulfilled as well. □

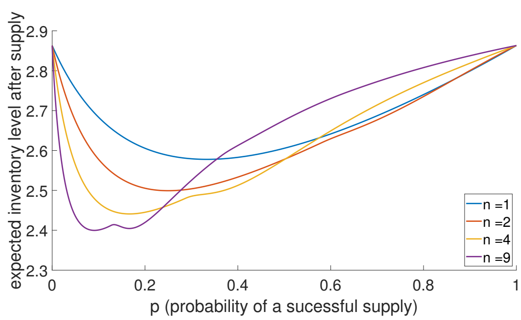
For the analysis we use the same parameters as in (3.3). We also use four different parameters for n, which are $n=1$ (this is equivalent to a Bernoulli distribution), $n=2$, $n=4$ and $n=9$. As we can see in Figure 3.4 the average cost for $p=0$ and $p=1$ are the same for all n, since all of these models reduce to the same deterministic one. There is no uncertainty about the supply the only difference is the amount of the supply. So for all models the lower threshold (cf. Figure 3.4(a)) for $p=0$ and $p=1$ is the same and the upper threshold (cf.



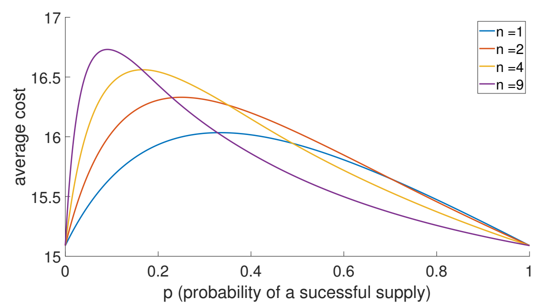
(a) lower threshold



(b) upper threshold



(c) expected inventory level after supply [1]



(d) average cost

Figure 3.4.: p-dependence in a drifted Brownian motion inventory model with binomial distributed supply ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ c_h = 3,\ c_b = 4$)

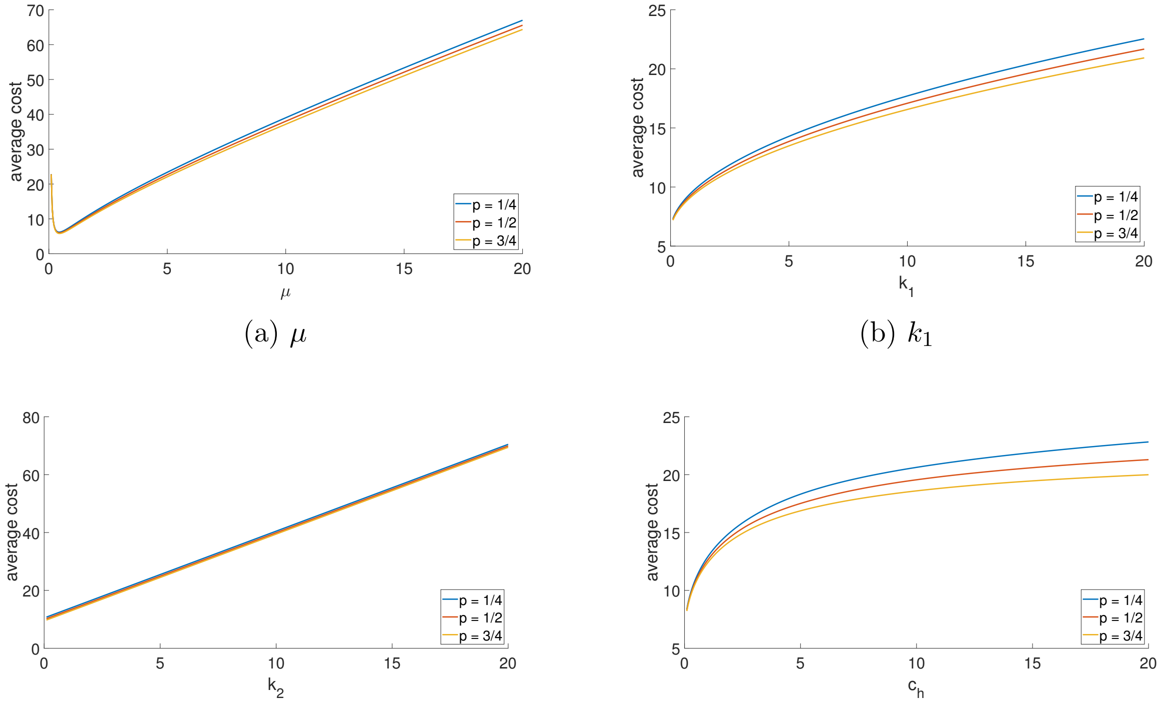
Figure 3.4(b)) is chosen in a way that the actual order size in the case $p = 0$ equals the order size at $p = 1$. So for $n = 1$ the order size at $p = 0$ is the double the order size at $p = 1$, for $n = 2$ it is triple the order size and in general for $n = k$ it is $k + 1$th times the order size. Furthermore we see that the p, which maximizes the average cost (cf. Figure 3.4(d)) is the greater the smaller n is. For all four n the average costs increase close to zero for increasing p until they reach their maximum between $p = 0.1$ $(n = 9)$ and $p = 0.35$ $(n = 1)$. After that they all decrease until they reach their minimum at $p = 1$ (which is as explained above also reached for $p = 0$). The greater the value of n the grater is also the maximum average cost. On the other hand, the average cost are also the cheaper the closer p is to one. As we can see in Figure 3.4(c) the upper threshold is chosen in a way that the expected inventory level is always below the supply in a deterministic model. This strategy avoids large holding costs through unexpectedly high orders, which is also the reason why the curve decreases more

[1] given by $\sum_{k=0}^{n} p_k x_{k,y,z} = \frac{(1-p)ny+(1+pn)z}{n+1}$

for greater n. However, with a smaller expected supply, ordering occurs more frequently. Therefore the average costs increase very quickly close to $p = 0$. On the other hand if p moves closer to one it is an advantage having a great n, since the probability of a small supply is much smaller than for small n. Therefore the expected value for larger n is for p closer to one greater than for small n, with a large decrease of the order size close to zero (especially for large n), which flattens out close to one. The difference between the different n also decreases quickly and for $p = 0.7$ there is already only a very small difference between all four n. The lower threshold, in contrast, stays relative stable. The highest deviation is 0.35 for $n = 9$. However on this small change level the curves look very similar to the average cost curves. But especially for $n = 4$ and $n = 9$ the curve has more dents.

In Figure 3.5 we see the results of a sensitivity analysis with regard to μ, k_1, k_2 and c_h for $p = 0.25$, $p = 0.5$ and $p = 0.75$. The plots show the different average cost level in a model with the same cost structure as above, except for the corresponding variable parameter. In this case $n = 4$, meaning there are five possible supply levels. As we would expect from our previous analysis and Figure 3.4, the costs for $p = 0.25$ are the highest, while they are the lowest for $p = 0.75$. All in all, the plots look very similar as Figure 3.2. For μ (cf. Figure 3.5(a)) there is a steep decrease close to zero with a minimum at around $\mu = 0.3$. After that there is a nearly linear increase with only very small differences between the different p. The high costs close to zero arise from the high holding costs in a model with nearly no drift, while the increase after the local minimum comes from the more frequent orders.

For k_1 (cf. Figure 3.5(b)) the increase close to zero is a bit steeper than for greater k_1. Furthermore, we see that for small k_1, there is nearly no difference in the average costs for the different p, while with increasing k_1 also the difference between the average costs for the different p increases slowly. The reason here is that for small k_1 more frequent orders are only slightly more expensive, since the order consists from the fixed cost k_1 and the cost for the actual supply. However with the increasing fixed costs it gets more expensive to order more frequently and for small p there is a tendency for more orders.



(a) μ (b) k_1

(c) k_2 (d) c_h

Figure 3.5.: Sensitivity analysis of the average cost in a drifted Brownian motion inventory model with binomial distributed supply for different model parameters ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ c_h = 3,\ c_b = 4$, if not variable)

For k_2 (cf. Figure 3.5(c)) there is linear increase, since the two thresholds are constant in this case. The explanation is the same as in the case of a uniform distributed supply.

For c_h (cf. Figure 3.5(d)) there is rapid increase in the average costs close to zero, which flattens out for increasing c_h. Again close to zero there is only a very small difference between the different p, which increases for larger c_h. This is the case, since for very small c_h the orders get very large, as it is very cheap to have high inventory levels and there are only few orders. Because of this for small p the thresholds become very large, since even in the case of a unexpectedly large supply the holding costs are still relatively cheap. All in all there are fewer orders, so the random supply does not play in the average cost. However, for greater c_h there are many more orders, since now the holding costs are very expensive and causes the randomness of the supply to play a more important role in the expected average cost.

3.1.3. Beta distribution

Let us now assume that the supply is beta-distributed on $[p_a, z]$ with parameters p and q. Most books call this the generalized beta distribution (while the beta distribution is defined on $[0, 1]$). This means

$$\mathcal{Q}(dv, y, z) = c\mathbb{1}_{\mathcal{L}^a}(v-a)^{p-1}(b-v)^{q-1}\mathrm{d}v,$$
$$c = \frac{\Gamma(p+q)(b-a)^{p+q-1}}{\Gamma(p)\Gamma(q)}$$

Let X be a beta distributed random variable with parameters p and q. Then

$$Y = (b-a)X + a$$

is a generalized beta distributed random variable on (a, b) with the same parameters. Similar to the proof of Theorem 3.2) we can prove that the beta distribution also fulfills Condition 2.2. However, with this more complicated density it is not possible anymore to calculate any simple antiderivatives for c_1, g_0 and ζ, so we have to use numerical approximations here.

For the analysis we set our parameters similar to (3.3):

$$\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ c_h = 3,\ c_b = 4,\ p = 2,\ q = 5.$$

In Figure 3.7(d) we see that the cost structure in this case is very similar to the one of the uniform supply (cf. Figure 3.1(d)) with quick decrease close to $a = 0$, which flattens out for values of a near one. This is also true for the upper threshold (cf. Figure 3.7(b)), which is very similar as in Figure 3.1(b). The expected supply after an order (cf. Figure 3.7(c)), which is given by $(z - p_a)\frac{p}{p+q} + p_a$, again is very similar to the one of the uniform distribution (cf. Figure 3.1(c)). In both cases the lower threshold (cf. Figure 3.1(c)) & Figure 3.1(a)) is relatively constant (even if the curves look different, with a monotone decrease in the case of the beta distributed supply). All in all the behavior of this beta distribution is very

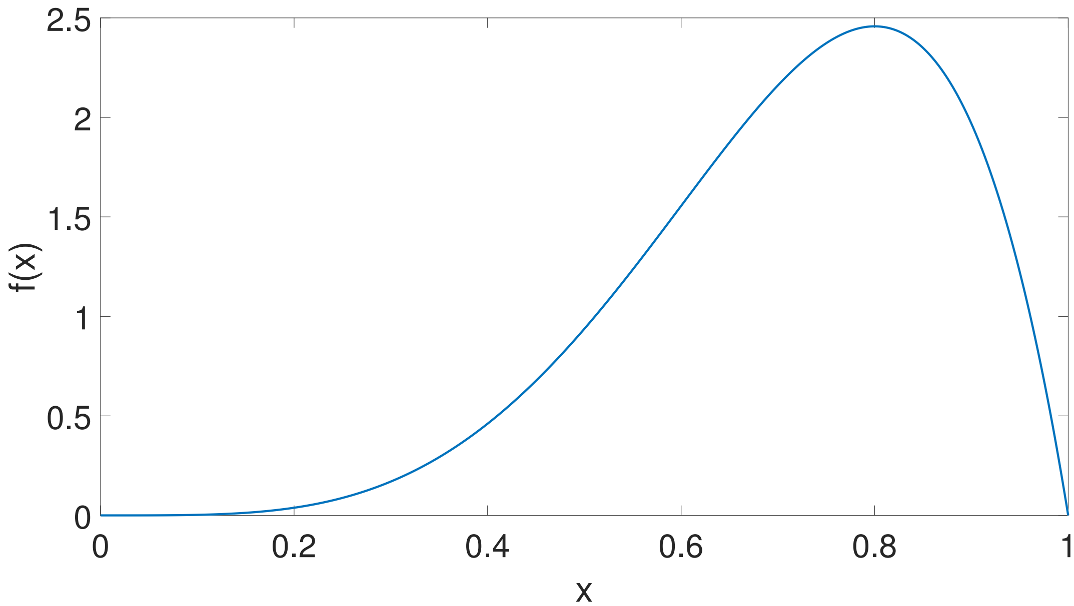
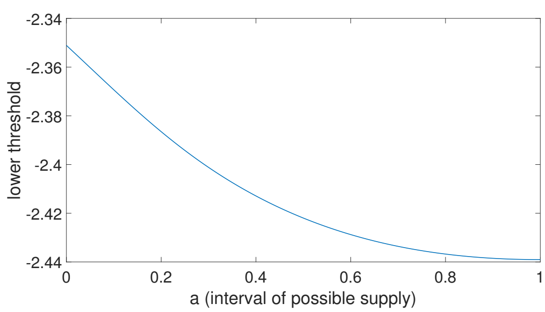
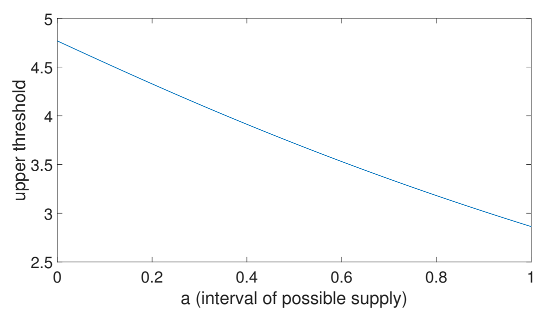


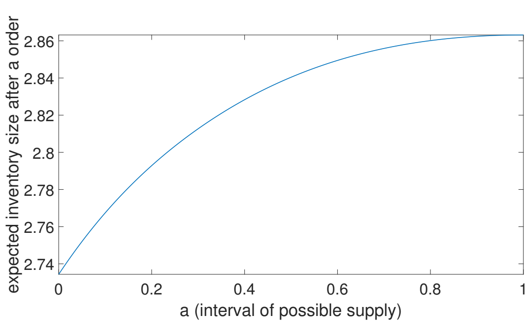
Figure 3.6.: Probability density function of a standard beta distribution (on $[0, 1]$) with parameters $p = 2$ and $q = 5$



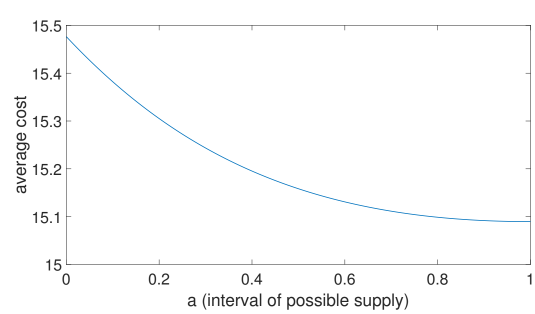
(a) lower threshold



(b) upper threshold

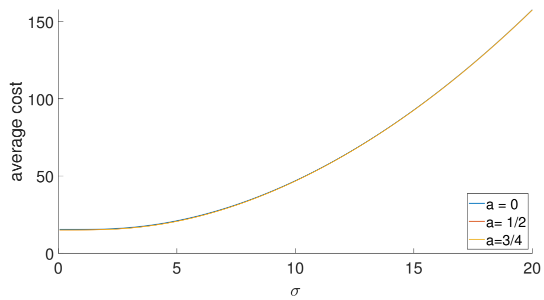


(c) expected inventory level after supply

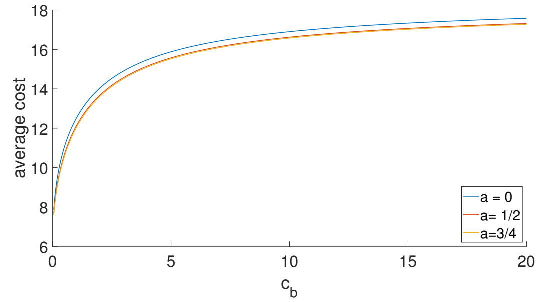


(d) average cost

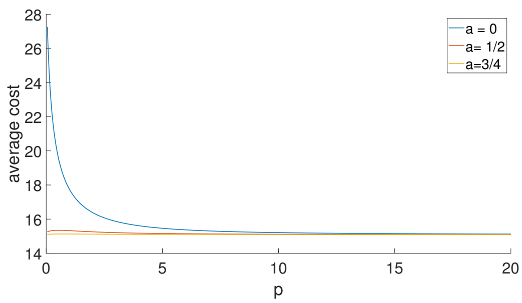
Figure 3.7.: a-dependence in a drifted Brownian motion inventory model with beta distributed supply ($\mu = 3; \sigma = 1; k_1 = 8; k_2 = 2; c_h = 3; c_b = 4, p = 2, q = 5$)



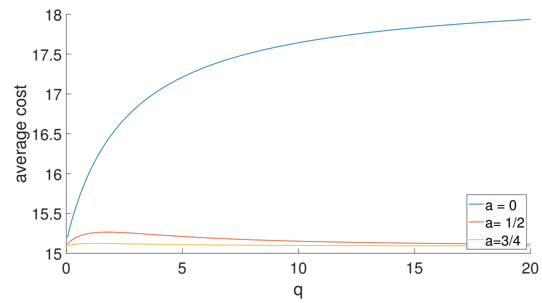
(a) σ



(b) c_b



(c) p



(d) q

Figure 3.8.: Sensitivity analysis of the average cost in a drifted Brownian motion inventory model with beta distributed supply for different model parameters ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ c_h = 3,\ c_b = 4,\ p = 2,\ q = 5$, if not variable)

comparable to the one of the uniformly distributed supply. On the one hand in this beta distribution there is less probability mass on the lower threshold and so less probability to get very low supplies. However, on the other hand there is also a smaller probability to get a almost full supply. As a result this two effects almost cancel each other so that as a result the both distributions looks very similar.

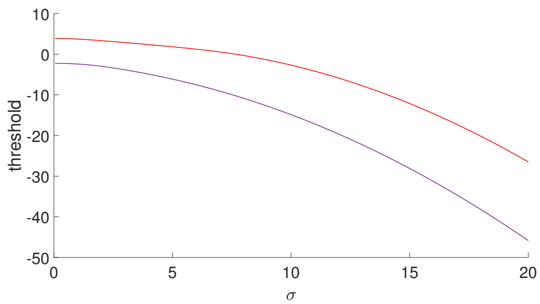
We now continue with a sensitivity analysis of different model parameters of the average cost (cf. Figure 3.9) and the thresholds (cf. Figure 3.9) for different a. We start with σ, where the average costs (cf. Figure 3.8(a)) are increasing very quickly. This is very natural, since for greater σ the fluctuations are much higher. This leads on the one hand to much higher (back-)holding costs due to the higher inventory levels and on the other hand to higher ordering costs, since the time between orders decreases on average. In Figure 3.9(a) we can see the thresholds both decrease for increasing σ. However the difference between

the upper threshold and the lower threshold increases a bit. Interestingly both thresholds are smaller than zero for $\sigma > 10$. This result is not very intuitive, since the back-holding costs are higher than the holding costs in this model.

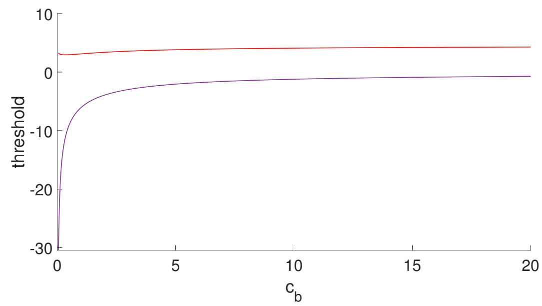
For c_b (factor of the back-holding costs) the average costs (cf. Figure 3.8(b)) are rapidly increasing for c_b close to zero. The curve flattens out for $c_b > 2$. A similar behavior can be seen for the thresholds (cf. Figure 3.9(b)) with a quick increase of the lower threshold and a almost constant upper threshold for $c_b \in [0, 1]$. The explanation for this is very intuitive. For very small c_b it is very cheap to have a negative inventory level, since there are almost no back-holding costs. So in this case the goal is to have only negative inventory level and order very rarely. In contrast for large c_b it is very expensive to have negative inventory levels and therefore the lower threshold increases so that is close to zero. The average cost curve flattens out, since it is tried to avoid negative inventory levels and therefore the back-holding costs do not play an important role if c_b is large.

For p, the average cost (cf. Figure 3.8(c)) is almost constant for $a = \frac{1}{2}$ and $a = \frac{3}{4}$. However for $a = 0$ the costs are decreasing rapidly between $p = 0$ and $p = 3$. The reason for this is that for small p there is high probability mass on the lower threshold. This means in the case of $a = 0$ the supplies of the orders are close to zero. This leads to very frequent orders and therefore high costs through the order fixed costs. For $a = \frac{1}{2}$ and $a = \frac{3}{4}$ this problem does not occur, since there is a guaranteed minimum supply. We can see that also in the upper threshold (cf. Figure 3.9(c)), which is decreasing for p close to zero.

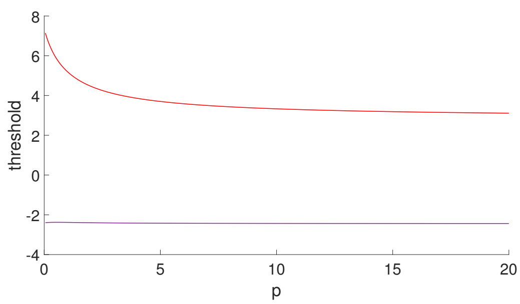
A different cost structure can be seen for q (cf. Figure 3.8(d)). Even if for $p = \frac{1}{2}$ and $p = \frac{3}{4}$ the cost function again is almost constant, there is increase for $a = 0$ with increasing q. However in comparison to the model parameter p the increase is much slower. The reason for this behavior is that for increasing q there tends to be more probability mass on the lower threshold. In Figure 3.9(d) we can see that the lower threshold is nearly constant, while the upper threshold increases. Again the reason is to hold the expected inventory level after supply on a relatively constant level.



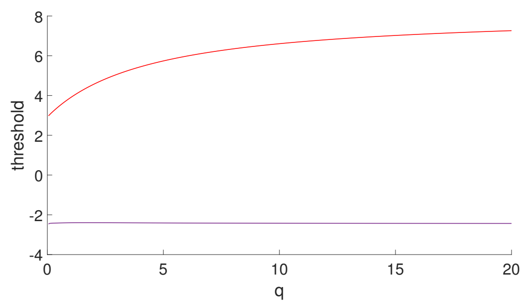
(a) σ



(b) c_b



(c) p



(d) q

Figure 3.9.: Sensitivity analysis of the thresholds in a drifted Brownian motion inventory model with beta distributed supply for different model parameters ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ c_h = 3,\ c_b = 4,\ p = 2,\ q = 5$, if not variable)

3.2. Drifted Brownian motion with reflection at 0

In this section we define we define the inventory model

$$\mathrm{d}X_0(t) = -\mu \mathrm{d}t + \sigma \mathrm{d}W(t) + \mathrm{d}L_t(0),\ X_0(0) = x_0 \in [0, \infty).$$

with $\mu, \sigma > 0$, W a standard Brownian motion, and L_t the local time process at 0 of X. This is a Brownian motion with reflection at $\{0\}$. A result of this is that the generator of X_0 is the same as for a normal drifted Brownian motion. For a detailed discussion of this topic see [1].

For our model we assume that the holding cost $c_0 : [0, \infty) \to \mathbb{R}^+$ and the the ordering costs $c_1 : \{(y, z) : 0 \leq y \leq z\}$ are given by

$$c_0(x) = k_3 x + k_4 e^{-x},\ c_1(y, z) = k_1 + k_2\sqrt{z - y},$$

where $k_1, k_2, k_3, k_4 > 0$. This means the costs for having a very small inventory are bounded, while the order function is concave, which means that savings due to economics of scale are possible.

Theorem 3.1

In the model above $g_0(x)$ and $\zeta(x)$ are given by

$$g_0(x) = \hat{g}_0(x) - \hat{g}_0(x_0)$$
$$\zeta(x) = \frac{x - x_0}{\mu}$$

with

$$\hat{g}_0(x) = \frac{k_3}{2\mu}x^2 + \frac{k_3\sigma^2}{2\mu^2}x + \frac{2k_4}{2\mu + \sigma^2}\left(1 - e^{-x}\right)$$

Proof. Since the generator A is the same as for a regular drifted Brownian motion $\zeta(x)$ is

the same as in the last chapter. For g_0 we calculate

$$
\begin{aligned}
-\mu g_0'(x) &= -k_3 x - \frac{k_3\sigma^2}{2\mu} - \frac{2\mu k_4}{2\mu+\sigma^2}e^{-x} \\
\frac{\sigma^2}{2} g_0''(x) &= \frac{k_3\sigma^2}{2\mu} - \frac{\sigma^2 k_4}{2\mu+\sigma^2}e^{-x}
\end{aligned}
$$

So $Ag_0 = -c_0(x)$, which finishes the proof. □

3.2.1. Uniform distribution

In this section we assume the same distribution as in subsection 3.1.1. So we get

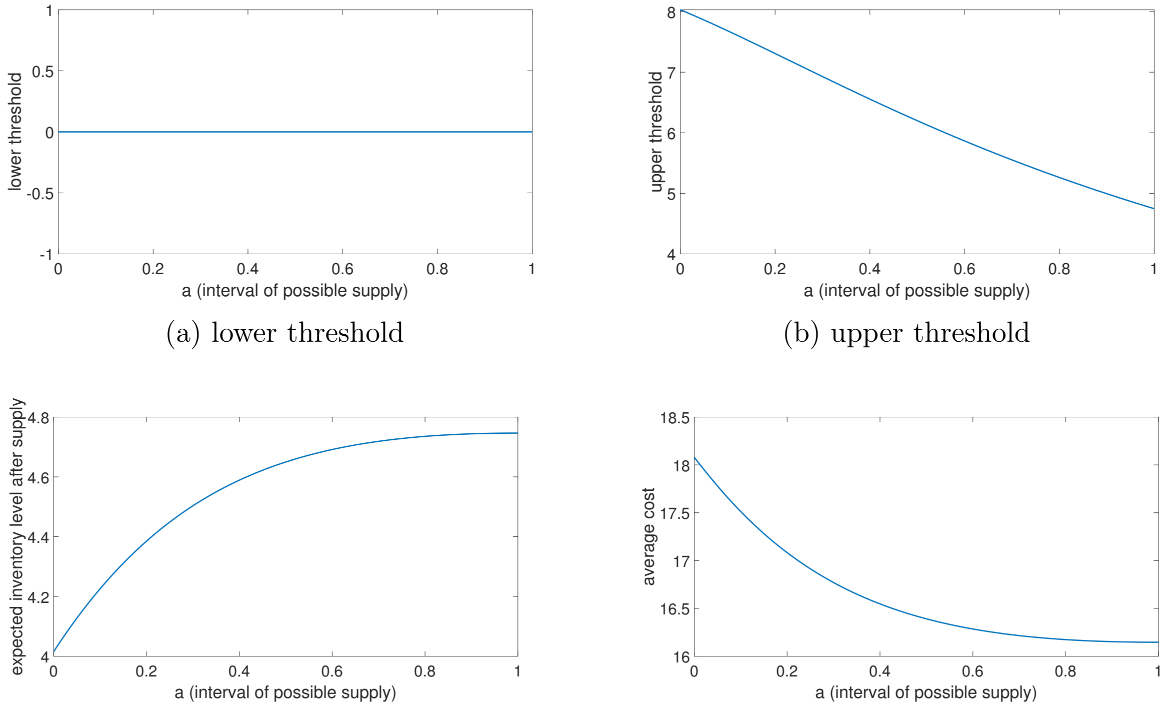
$$
\begin{aligned}
\widehat{c}_1(y,z) &= \int_{p_a}^{z} \frac{k_1 + k_2\sqrt{v-y}}{q_a}\mathrm{d}v = \\
&= k_1 + \frac{2k_2}{3q_a}\left(\sqrt{(z-y)^3} - \sqrt{(a(z-y))^3}\right) \\
&= k_1 + \frac{2k_2}{3(1-a)}\left(\sqrt{z-y} - a\sqrt{a(z-y)}\right), \\
\int_{p_a}^{z} g_0(v)\mathrm{d}v &= \frac{k_3\left(z^3-p_a^3\right)}{6\mu q_a} + \frac{k_3\sigma^2\left(z^2-p_a^2\right)}{4\mu^2 q_a} + \frac{2k_4}{2\mu+\sigma^2}\left(1 + \frac{e^{-z}-e^{-p_a}}{q_a}\right) \text{ and} \\
\int_{p_a}^{z} \zeta(v)\mathrm{d}v &= \frac{1}{\mu}\left(z^2 - p_a^2\right).
\end{aligned}
$$

This leads to

$$
\begin{aligned}
\widehat{B}_{\mathcal{Q}} g_0(y,z) = &\frac{k_3}{2\mu}\left(\frac{z^3-p_a^3}{3q_a} - y^2\right) + \frac{k_3\sigma^2}{2\mu^2}\left(\frac{z^2-p_a^2}{2q_a} - y\right) \\
&+ \frac{2k_4}{2\mu+\sigma^2}\left(2 + \frac{e^{-z}-e^{-p_a}}{q_a} - e^y\right)
\end{aligned}
$$

For the analysis we establish the following parameters

$$
\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ k_3 = 3,\ k_4 = 4. \tag{3.4}
$$



(a) lower threshold

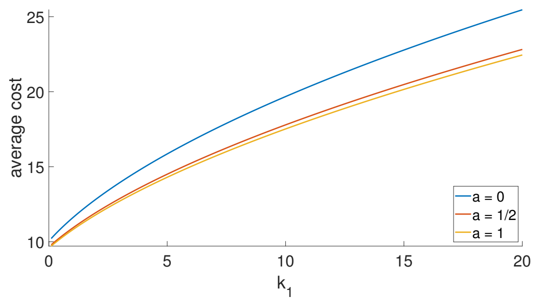
(b) upper threshold

(c) expected inventory level after supply

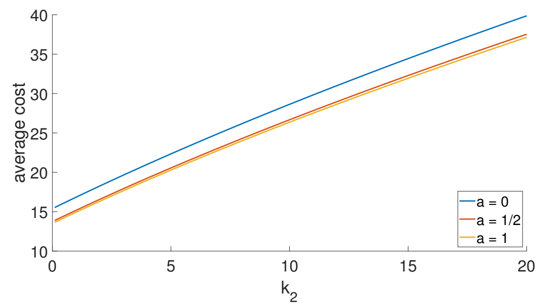
(d) average cost

Figure 3.10.: a-dependence in a drifted Brownian motion with reflection at zero inventory model with uniformly distributed supply ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ c_h = 3,\ c_b = 4$)

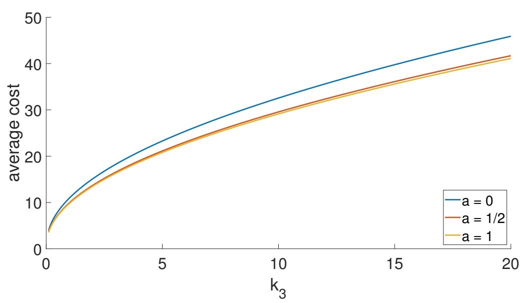
Figure 3.15(a) shows that the lower threshold is zero for all $a \in [0, 1]$, which can be explained by the low costs for small inventory (k_4). This means that the upper threshold (cf. Figure 3.10(b)) and the amount of order are the same. They decrease linearly by around 40% between $a = 0$ and $a = 1$. As a result, the expected order supply (cf. Figure 3.10(c)) increases a little bit for increasing a. The reason for this is that for $a = 0$ to get in the mean a supply close to 6, the order size must be double the amount of the goal of the supply. However in this case there is a risk of getting a nearly complete order, which increases the holding cost. On the other hand a too small order size increases the risk of getting a very small order. This would lead to only a small time interval between two orders, since the inventory will decrease quickly to zero again and therefore to double payment of the order fixed costs. The average (cf. Figure 3.10(d)) cost decrease at around 10% from 24 to around 21.5. This can be explained with the fact of the missing variation in the supply an therefore



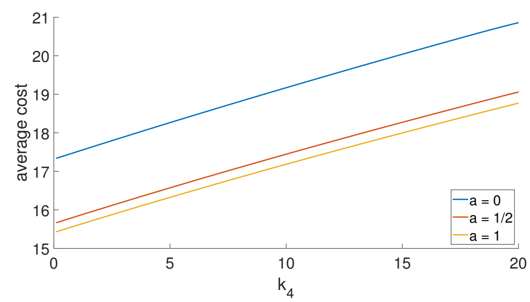
(a) k_1



(b) k_2



(c) k_3

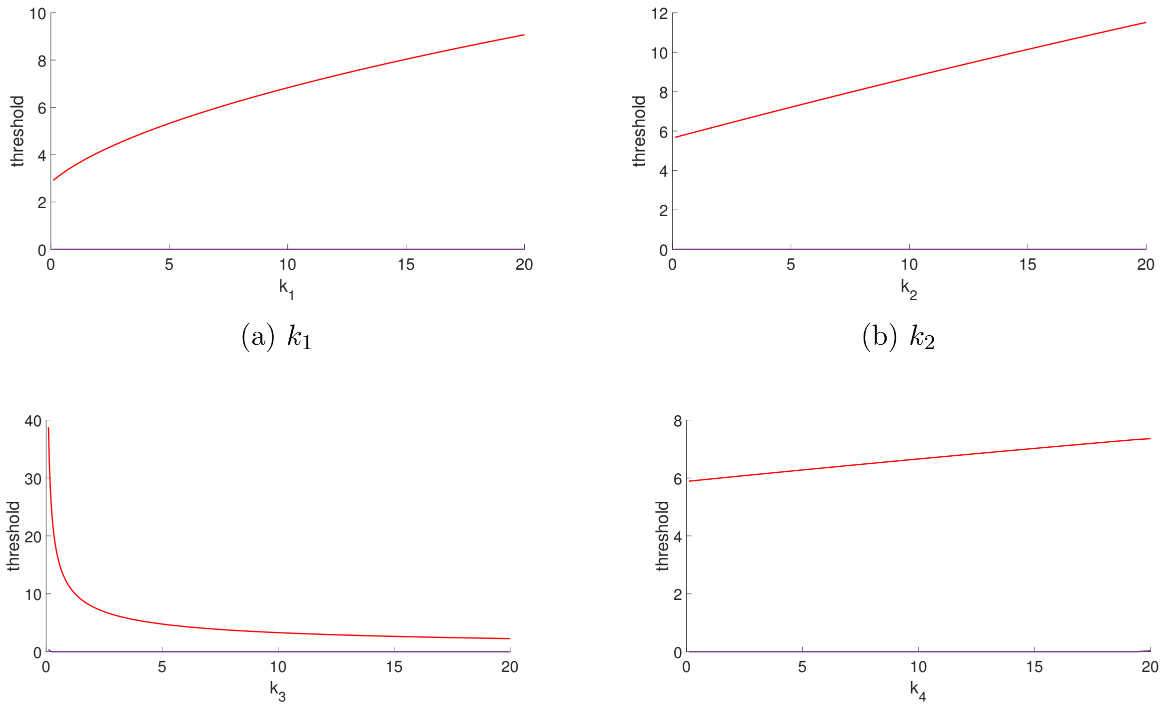


(d) k_4

Figure 3.11.: Sensitivity analysis of the average cost in a drifted Brownian motion with reflection at zero inventory model with uniformly distributed supply for different model parameters ($\mu = 3$, $\sigma = 1$, $k_1 = 8$, $k_2 = 2$, $k_3 = 3$, $k_4 = 4$, if not variable)

the stable costs. Another observation is that the average cost decreases faster for small a, while the decrease is very slow for $a > 0.6$. In practice this means that if the guaranteed supply amount would be 60% of the order size, the long term average price for the inventory only increases by a small amount.

The next step is a sensitivity analysis of the average cost (cf. Figure 3.11) and the threshold (cf. Figure 3.12) for the four different cost parameter (k_1-k_4). Unsurprisingly all cost curves are monotonously increasing. For k_2 (cf. Figure 3.11(b)) and k_4 (cf. Figure 3.11(d)) the average cost curve looks linear, with a bit steeper increase for k_2. The same can be said for the upper threshold (cf. Figure 3.12(b) & Figure 3.12(d)), while the lower threshold is constant zero. Since k_2 is the price per unit (however with possible economics of scale) an increase in k_2 leads to higher orders to profit from the possible savings for high orders. These



(a) k_1 (b) k_2 (c) k_3 (d) k_4

Figure 3.12.: Sensitivity analysis of the thresholds in a drifted Brownian motion inventory model with reflection at zero and uniformly distributed supply for different model parameters ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ k_3 = 3,\ k_4 = 4, a = \frac{1}{2}$, if not variable)

get more important for higher k_2. However this leads also to higher holding costs. In the case of k_4, which is the parameter for the penalty cost for small inventory, the increase is just a result of the higher costs close to the inventory level zero. Although the upper threshold increases a bit, to increase the expected inventory level after supply and therefore the time between two orders (at the point zero). However, this effect is very small. We also see that the for high k_4 the lower threshold is still at zero, so it is still cheaper to wait until the inventoory level has reached its minimum and then order at this point, instead of ordering before and therefore more often.

For k_1 the average cost (cf. Figure 3.11(d)) looks almost linear, even if it curve flattens out a little bit for greater k_1. The same holds for the upper threshold, while the lower threshold is constant zero. This case is very intuitive, since k_1 are the order fixed costs. With these costs

increasing the goal is to avoid frequent orders and therefore increase the amount of order. The reason that the curve flattens out is that for higher orders there is a small economics of scale effect due to the higher orders.

Lastly for k_3 the average costs are increasing quickly for k_3 close to zero and flatten out with increasing k_3. On the other hand the upper threshold decreases very rapidly for $k_3 \in [0, 2]$ and then flattens out. Again this is very intuitive, since for small k_3 there are almost no holding costs (at least for inventory levels not close to zero), so the order sizes become very big to profit from this and increase the time between two orders. However this effect decreases with a rate of $\frac{1}{k_3}$ and for greater k_3 it is very expensive to have high inventory levels, so the upper threshold becomes very small.

3.2.2. Polynomial distribution

In this section we will take a look at a generalization of the model of the previous section. Again the nominal delivery is randomly distributed on $[(1-a)y + az, z]$ ($a \in [0, 1]$), but this time the kernel distribution is given by

$$\mathcal{Q}_k(\mathrm{d}v, y, z) = c\mathbb{1}_{((1-a)y+az,z)}(v)v^k\mathrm{d}v,$$

where

$$c = \begin{cases} \frac{k+1}{z^{k+1}-[(1-a)y+az]^{k+1}} & k \neq -1 \\ \frac{1}{\ln z - \ln[(1-a)y+az]} & k = -1, \end{cases} \tag{3.5}$$

with $k \in \mathbb{R}$. For $k = 0$ we get the uniform distribution from above. For $k > 0$ we get a higher probability of getting what we ordered on the other hand for $k < 0$ we have a higher

chance of getting the lower bound of the interval. We calculate

$$\widehat{c}_1(y,z) = k_1 + ck_2 \int_{p_a}^{z} v^k \sqrt{v-y}\ \mathrm{d}v,$$

which does not have an antiderivative. We further compute

$$\widehat{B}_{\mathcal{Q}}\zeta(y,z) = \frac{c}{\mu}\left(\int_{p_a}^{z} v^{k+1}\mathrm{d}v - y\right) = \begin{cases} \frac{c\left(z^{k+2}-p_a^{k+2}\right)}{(k+2)\mu} - \frac{y}{\mu} & k \neq -2 \\ \frac{c(\ln z - \ln p_a) - y}{\mu} & k = -2 \end{cases}$$

and $g_0(y,z) =$

$$\begin{cases} c\left(\frac{k_3(z^{k+3}-p_a^3)}{2(k+3)\mu} + \frac{k_3\sigma^2(z^{k+2}-p_a^{k+2})}{2(k+2)\mu} + \frac{2k_4}{2\mu+\sigma^2}\left(1-\int_{p_a}^{z} e^{-v}v^k\mathrm{d}v\right)\right) & k \neq -2,-3 \\ c\left(\frac{k_3(z^{k+3}-p_a^3)}{2(k+3)\mu} + \frac{k_3\sigma^2(\ln z - \ln p_a)}{2\mu} + \frac{2k_4}{2\mu+\sigma^2}\left(1-\int_{p_a}^{z} e^{-v}v^k\mathrm{d}v\right)\right) & k = -2 \\ c\left(\frac{k_3(\ln z - \ln p_a)}{2\mu} + \frac{k_3\sigma^2(z^{k+2}-p_a^{k+2})}{2(k+2)\mu} + \frac{2k_4}{2\mu+\sigma^2}\left(1-\int_{p_a}^{z} e^{-v}v^k\mathrm{d}v\right)\right) & k = -3, \end{cases}$$

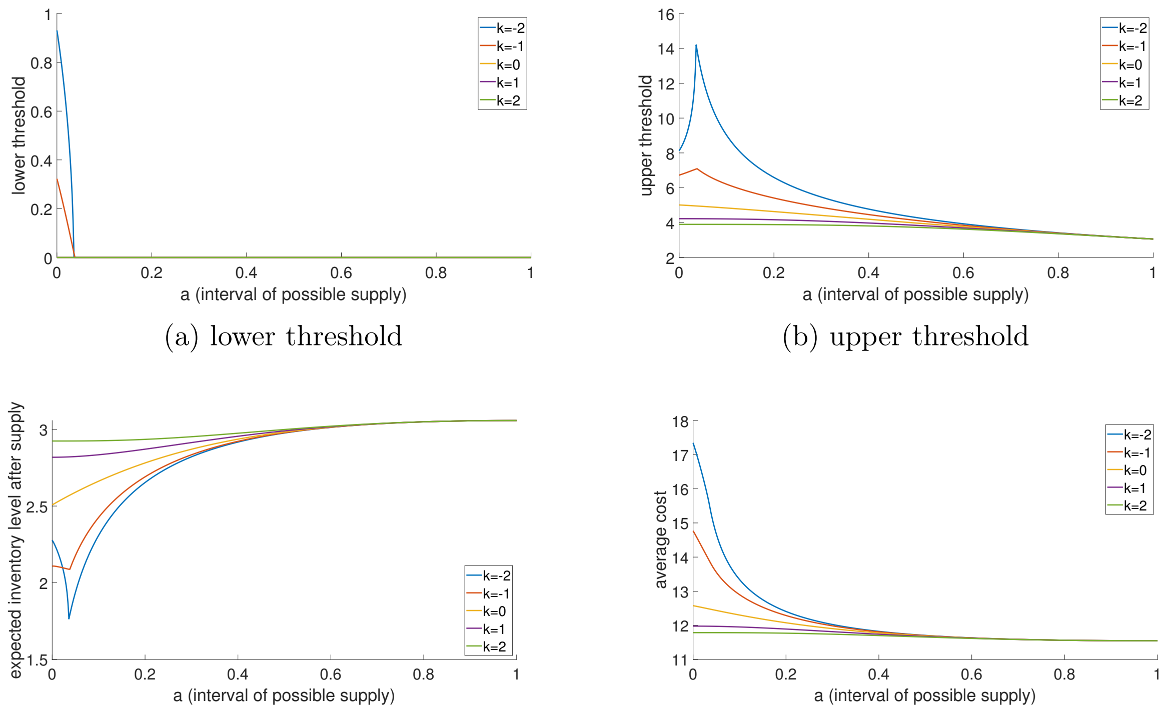
where the last integral does not have an explicit solution.

Theorem 3.2

The polynomial supply distribution fulfills Condition 2.2

Using the same parameters as in (3.4), Figure 3.13(d) shows that the average cost for different k only differs significantly for $a \in [0, 0.3]$. However in this interval there is a large difference between the k. While for $k \geq 0$ the average costs are nearly constant, the cost for $k < 0$ increases quickly for decreasing a. If $k \geq 0$ there is a high chance even for small a that the supply is close to the ordered amount, so there is only little cost for the randomness of the supply. The same holds for all k if $a \geq 0.3$, which is also shown in the bottom right plot, where even for $a = 0.25$ there is only a small deviation between $k \in [-5, 5]$. For very small

[2] given by $\frac{c(z^{k+2}-p_a^{k+2})}{k+2}$ for $k \neq -2$ and $c(\log z - \log p_a)$ for $k = 2$



(a) lower threshold (b) upper threshold

(c) expected inventory level after supply [2] (d) average cost

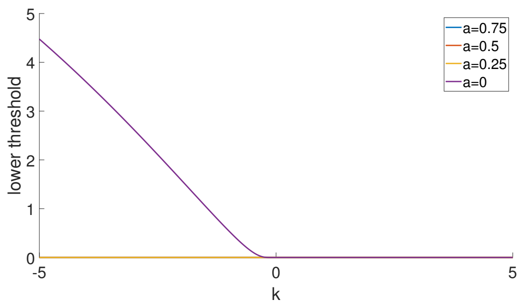
Figure 3.13.: a-dependence in a drifted Brownian motion with reflection at zero inventory model with polynomial distributed supply ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ k_3 = 3,\ k_4 = 4$)

k there is very much probability mass on the lower p_a. This means almost all the time there will be a supply, which is very close to p_a, that makes the supply almost deterministic. As a result the costs for $k = -5$ is smaller than for $k = -2$. If however, a is also very small this method lead to nearly no supply most of the times. As a consequence there are much more frequent orders and the fixed order cost lead to increasing average costs. This is shown in the steep decrease of the average costs for for $k = -2$ and $k = -1$ close to zero.

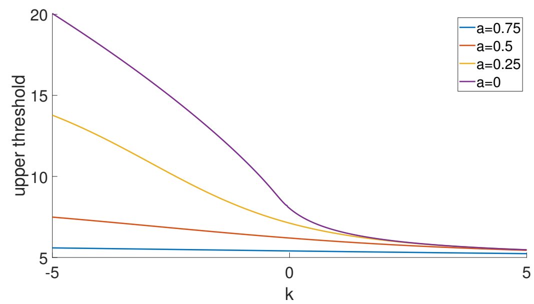
The lower threshold (cf. Figure 3.13(a)) is zero for all k and all a besides for very small a (between 0 and 0.05), where the threshold is not zero, since there are a lot of very small supplies and it is cheaper to pay not almost permanently the penalty of k_4 for small inventories. The constant lower threshold as well as the the almost constant upper threshold (cf. Figure 3.13(b)) for $k = 0, 1, 2$ are again a result of the relative high certainty of the actual supply. For $k = -2$ there is however the high peak at $a = 0.02$ (there is also a very small peak for $k = -1$), which is also the point where the lower threshold is zero for the first time. As explained above for such a low k the expected inventory level after supply (cf. Figure 3.13(c)) is close to the lower threshold, so it is necessary to make big orders to avoid frequent orders due to very small supply, but this effect decreases very quickly again for increasing a since supplies close to zero become impossible.

In Figure 3.14 we can study the behavior of the model depending on k. For $a \neq 0$ the average cost (cf. Figure 3.14(d)) are relative stable with obviously the lowest cost for $a = 0.75$. Interestingly, the average cost for $a = 0.25$ increases first, before having the maximum value at around $k = -2$ and constantly decreasing for $k > -2$ (the same holds for $a = 0.5$ and $a = 0.75$, however on a much smaller level). The reason for this is that a model with very small k is comparable to a binomial distributed supply model with p very close to zero, where the probability of getting the low end of possible supplies is so high that the model is close to a deterministic one.

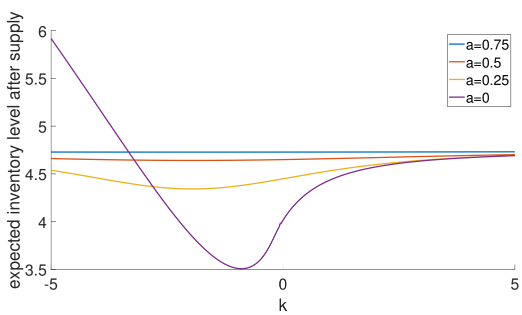
For $k \approx 2$, however there is enough mass to make larger supplies realistic enough that the upper threshold (cf. Figure 3.14(b)) decreases in way that the expected inventory level after



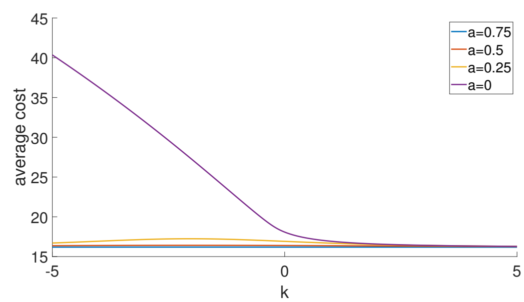
(a) lower threshold



(b) upper threshold



(c) expected inventory level after supply



(d) average cost

Figure 3.14.: k-dependence in a drifted Brownian motion with reflection at zero inventory model with polynomial distributed supply ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ k_3 = 3,\ k_4 = 4$)

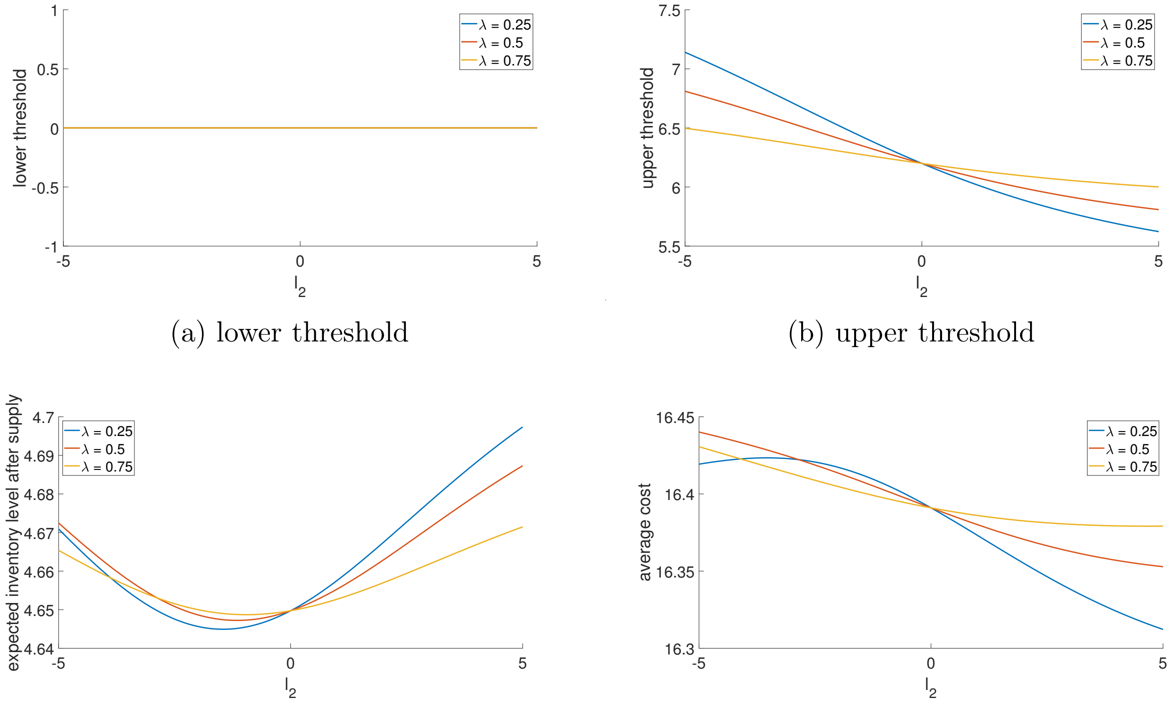
supply (cf. Figure 3.14(c)) decreases as well to avoid high holding costs through high supplies. This leads as a consequence to more frequent orders and, therefore, to higher costs. From that point, for increasing k, the average costs start to decrease since the probability mass is shifting to the upper threshold. For $k = 5$ the average cost differences for different a nearly vanished, because in this case all models are close to a deterministic with full supply. For $a = 0$ the average cost have a different structure with a steep increase for decreasing $k < 0$. In comparison to the other a, where a minimum supply is guaranteed for $a = 0$, a small k leads to a lot supplies which are close to zero so to avoid the holding costs to zero the lower threshold is greater than zero for $k < -0.5$ (and increasing for decreasing k). The upper threshold is chosen very high for small k. Nevertheless, for $k = -5$, the expected supply (the difference between expected inventory level after supply and lower threshold) is only a bit greater than one. This leads to a lot of orders in a relatively small time interval and therefore to high average cost due to the fixed order costs. With increasing k the lower and upper threshold both decrease, and with that the expected inventory level after supply, which reaches its minimum at $k = -1$ and increases afterwards. For $k > 0$ the deviation to the average costs of the other a is very small and tends to zero for $k \to \infty$.

3.2.3. Mixed polynomial distributions

We now want to analyze mixed polynomial, which are sums of two polynomial weighted distributions, namely

$$\mathcal{Q}(\cdot, y, z) = \lambda \mathcal{Q}_{l_1}(\cdot, y, z) + (1-\lambda)\mathcal{Q}_{l_2}(\cdot, y, z), \ \lambda \in [0,1]$$
$$\text{with } \mathcal{Q}_{l_i}(\mathrm{d}v, y, z) = c\mathbb{1}_{((1-a)y+az,z)}(v)v^{l_i}\mathrm{d}v, \ i = 1, 2$$

with c as in (3.5). This distribution obviously fulfills all conditions, since it is a linear combination of two polynomial distributions. The cost function as well is just a linear combination of the two polynomial cost functions, so there is no need to calculate anything



(a) lower threshold (b) upper threshold

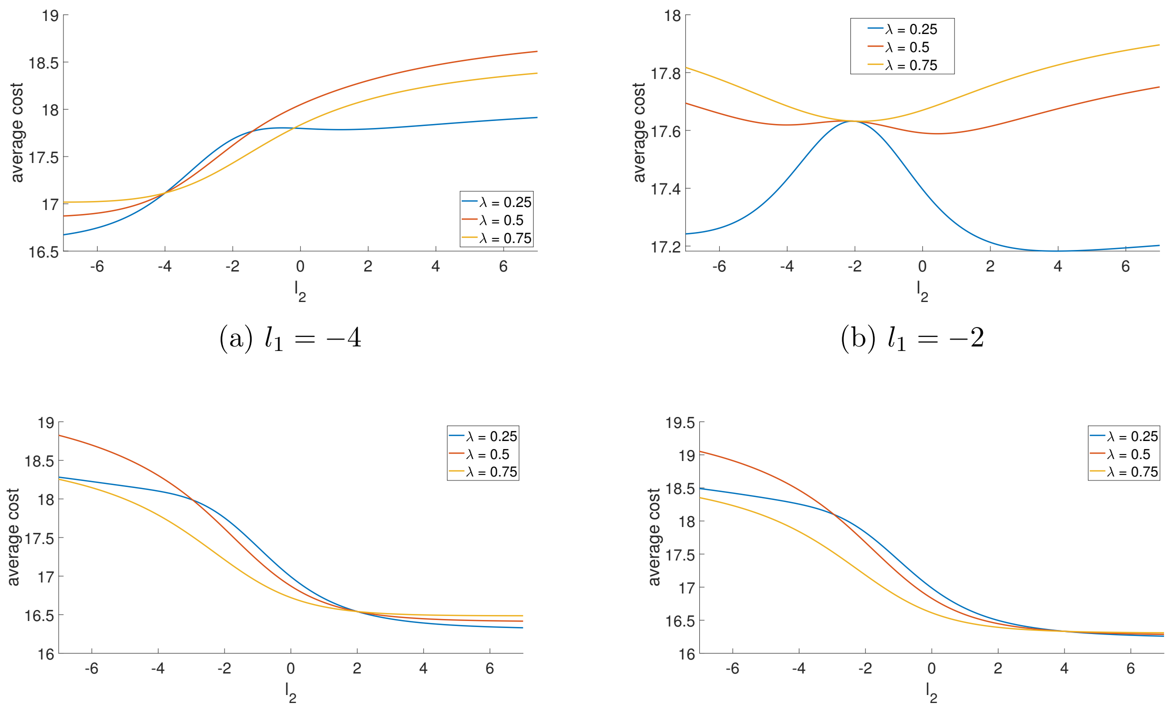
(c) expected inventory level after supply (d) average cost

Figure 3.15.: l_2-dependence in a drifted Brownian motion with reflection at zero inventory model with mixed polynomial distributed supply ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ c_h = 3,\ c_b = 4,\ l_1 = 0$)

new and we can directly start with the analysis. We set the following parameters

$$\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ k_3 = 3,\ k_4 = 4, a = \frac{1}{2}$$

In Figure 3.15(d) we see that the average costs for $l_1 = 0$ only varies a little for different l_2 and λ (the maximum deviation is below 0.9%). For $l_2 < 0$ the average cost are a bit higher than for the polynomial distribution with $k = 0$, while for $l_2 > 0$ (and especially $\lambda = 0.75$) the costs are below the polynomial distribution with $k = 0$. This can be explained by the fact that the expected inventory level after supply (cf. Figure 3.15(c)) can be chosen a little bit higher (the deviation here is around 1.2%) without risking too high holding costs. Especially for small λ, since in this case there is more weight on the l_2-distribution, where we saw in the last chapter that the average cost for polynomial distributions is decreasing



(a) $l_1 = -4$ (b) $l_1 = -2$

(c) $l_1 = 2$ (d) k_4

Figure 3.16.: Sensitivity analysis of the average cost in a drifted Brownian motion with reflection at zero inventory model with mixed polynomial distributed supply for different model parameters ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ k_3 = 3,\ k_4 = 4, a = \frac{1}{2}$)

for increasing k for $k > 0$.

For $l_2 < 0$ it is more complicated, while for $\lambda = 0.25$ and $\lambda = 0.5$ the cost increase relatively linearly for decreasing l_2, for $\lambda = 0.25$ increase until $l_2 \approx -3$ before decreasing again. As we saw in the subsection above in a polynomial distribution with $k = -5$ (and $a = 0.5$) the costs are a bit lower (on a small level) so with more weight on the l_2-distribution the costs are smaller than with more weight on the l_1-distribution ($\lambda = 0.25$). For $\lambda = 0.5$ we have the highest costs, because in this case the uncertainty in which distribution determines the supply is the highest. However all the above analyzed effects above are very small and the cost are very stable for the different l_2 and λ, which can be expected, since the average costs in a polynomial model with $a = \frac{1}{2}$ were very stable.

Nevertheless, we now want to analyze the average costs depending on l_2 for different l_1, which can be seen in Figure 3.16. For $l_1 = -4$ (cf. Figure 3.16(a)) we see that the average

costs are strictly increasing for increasing l_2, while for $l_1 = 2$ (cf. Figure 3.16(c)) and $l_1 = 4$ (cf. Figure 3.16(d)) the average costs are decreasing for increasing l_2. In the case of $l_2 = -2$ the cost structure is a bit more complicated with big differences for the different λ. Furthermore for these l_1 there is much more variation in the avearge costs than for $l_1 = 0$ (i.e. for $l_1 = 4$ the deviation is around 15%). This is because the maximum spread between the l_1 and l_2 is higher in these examples, which lead to higher variances and that normally leads to higher costs.

First we look at $l_1 = 2$ and $l_1 = 4$ where the average cost looks very similar with a flat decrease for $l_2 \in [-7, -2.5]$, that gets steeper until $l_2 = 2$ and then flattens out. Moreover the average cost curves of the single λ behave similarly. In the case of small l_2 the costs are the highest for $\lambda = 0.5$, while for $\lambda = 0.25$ and $\lambda = 0.75$ the cost are almost on the same level. As explained above the reason for this behavior is that in this case both the l_1 and the l_2 polynomial distribution is relative cheap. However with mixing them there is a high variance, which has it maximum for $\lambda = 0.5$ and therefore the highest costs. The reason that the decrease is relative small is that for increasing l_2 the decreasing costs due to the smaller variance are partly canceled by the the increasing cost of the polynomial distribution. That is also the explanation for the steep decrease in the interval $l_2 \in [-2.5, 0]$, where the average costs for a polynomial distribution decrease relative quickly. For $l_2 > 0$ both distributions now have most of the weight close to the upper threshold so the advantage of a smaller variance disappears. On the contrary the average cost decrease further even after the point $l_1 = l_2$, because both the l_1 and the l_2 polynomial distribution now tend to a supply close to the upper threshold and the the greater l_2 makes a great supply even more likely, which also explains that the smaller λ is the smaller the average cost, since a small λ means more weight on the l_2 distribution.

A very similar reasoning with a mirrored l_2-axis (since in this case the polynomial distribution has most of the weight on the lower threshold) can explain the average cost curve for $l_1 = -4$. The only difference is that for large l_2 the difference of the average costs for $\lambda = 0.25$ and

$\lambda = 0.75$ is greater with the cheapest cost for $\lambda = 0.25$. This is no big surprise, since in this case there is more weight on the cheaper l_2 polynomial distribution.

In the analysis of the polynomial distribution we saw that the average cost have a maximum at around $k = -2$, which can help us to understand the cost structure in the case of $l_1 = -2$. For $\lambda = 0.25$ the average costs have the maximum for $k = -2$, which means that in this case the higher costs through a higher variance is not only canceled but the cheaper costs for the strong weighted $(1 - \lambda = 0.75)$ l_2 polynomial distribution lead to savings. However for large $l_2 > 4$ the differences between the two distributions gets so large that the average cost increase again (even if it is only on a small level). However this effect is heavily dependent by a high weight on the cheaper distribution, so for $\lambda = 0.5$ though there is small decrease moving away from $l_2 = -2$ it is much smaller than for $\lambda = 0.25$ and it is not that long-lasting, since there is a decrease of the average cost in the interval $[-7, -4]$ resp. an increase in the interval $[0, 7]$. For further increased $\lambda = 0.75$ the behavior is kind of the opposite as for $\lambda = 0.25$ with a minimum at $l_2 = -2$. So in this case there is only a small weight on the cheaper l_2 polynomial distribution, which can not compensate the higher costs arising by the higher variance.

3.3. Geometric Brownian motion

In this section we assume that the inventory process is given by

$$\mathrm{d}X_0(t) = -\mu X_0(t)\mathrm{d}t + \sigma X_0(t)\mathrm{d}W(t),\ \ X(0) = x_0 \in (0, \infty),$$

with $\mu, \sigma > 0$ and W a standard Brownian motion. This means the inventory level in absence of orders is given by a geometric Brownian motion on $(0, \infty)$. The cost functions are $c_0 : (0, \infty) \to \mathbb{R}^+$ and $c_1 : \bar{\mathcal{R}} := \{(y, z) \in \mathbb{R}^2 : 0 < y \leq z\}$:

$$c_0(x) = k_3 x + k_4 x^\beta \text{ and } c_1(y, z) = k_1 + k_2\sqrt{z - y},$$

where $k_1, k_2, k_3, k_4 > 0$ and $\beta < 0$.

Theorem 3.1

In this model the functions g_0, ζ are given by

$$g_0(x) = \frac{k_3}{\mu}(x - x_0) - \frac{k_4}{\rho}(x^\beta - x_0^\beta) \text{ and } \zeta(x) = \frac{2}{2\mu + \sigma^2}(\ln x - \ln x_0),$$

with $\rho = \frac{\sigma^2\beta(\beta-1)}{2} - \mu\beta$.

Proof. We have $\hat{\mu}(x) = -\mu x$ and $\hat{\sigma}(x) = \sigma x$, so we get

$$\begin{aligned}
\hat{\mu}\zeta' + \frac{\hat{\sigma}^2}{2}\zeta'' &= \frac{2}{2\mu + \sigma^2}\left(\frac{-\mu x}{x} - \frac{\sigma^2 x^2}{2x^2}\right) = -1, \\
\hat{\mu}g_0' + \frac{\hat{\sigma}^2}{2}g_0'' &= \frac{-k_3\mu x}{\mu} - \frac{k_4\mu\beta x^\beta}{\rho} - \frac{k_4\beta(\beta-1)x^\beta}{2\rho} = -k_3 x - k_4 x^\beta = -c_0(x)
\end{aligned}$$

□

3.3.1. Uniform distribution

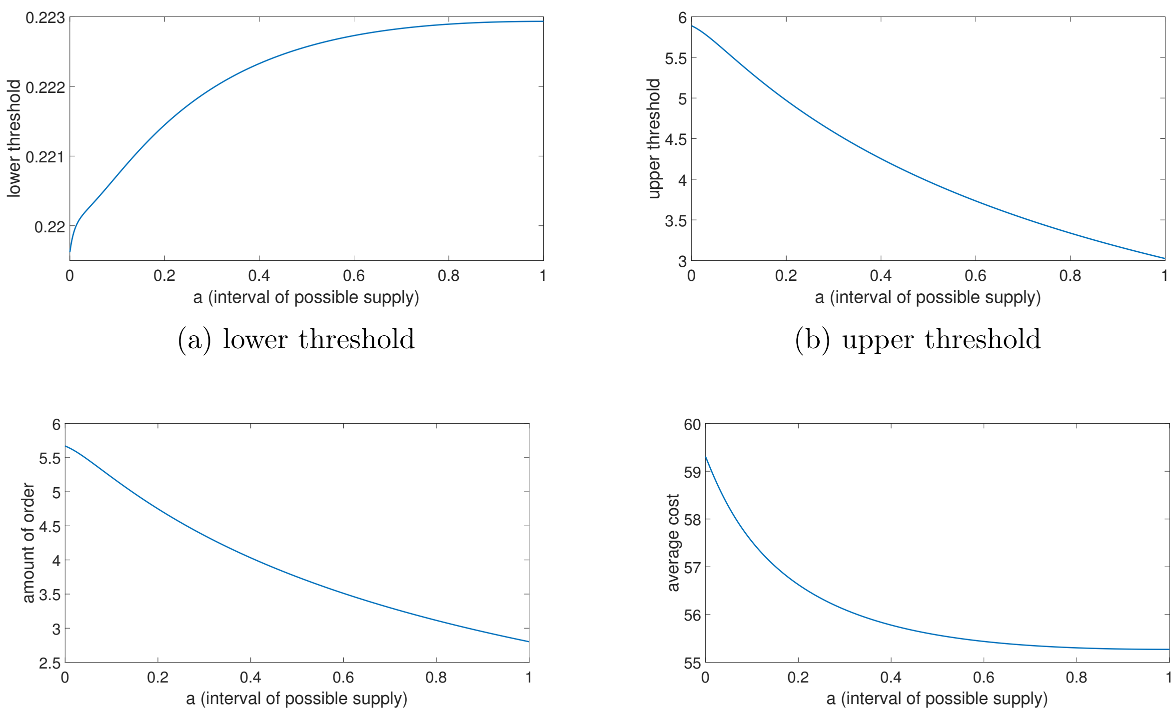
Let $\mathcal{Q}(\mathrm{d}v, y, z)$ be as in (3.2). Then we get with $p_a = (1-a)y + az$

$$\begin{aligned}
\widehat{c}_1(y, z) &= k_1 + \frac{2k_2}{3(1-a)}\left(\sqrt{z-y} - a\sqrt{a(z-y)}\right), \\
\widehat{\zeta}(y, z) &= \int_{p_a}^{z} \frac{2\ln v}{q_a(2\mu + \sigma^2)}\mathrm{d}v = \frac{1}{q_a(2\mu + \sigma 2)}\left(z\ln z + p_a - p_a \ln p_a - z\right) \\
\widehat{g}_0(y, z) &= \int_{p_a}^{z}\left(\frac{k_3 v}{\mu} - \frac{k_4 v^\beta}{\rho}\right)\mathrm{d}v = \frac{k_3(z^2 - p_a^2)}{2\mu} + \frac{k_4}{\rho}\cdot\begin{cases} \frac{z^{\beta+1} - p_a^{\beta+1}}{\beta+1} & \beta \neq -1 \\ \ln z - \ln p_a & \beta = -1. \end{cases}
\end{aligned}$$

For the analysis we set

$$\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ k_3 = 3,\ k_4 = 4,\ \beta = -2. \tag{3.6}$$

Figure 3.17 shows the lower threshold, the upper threshold, the amount of order (upper



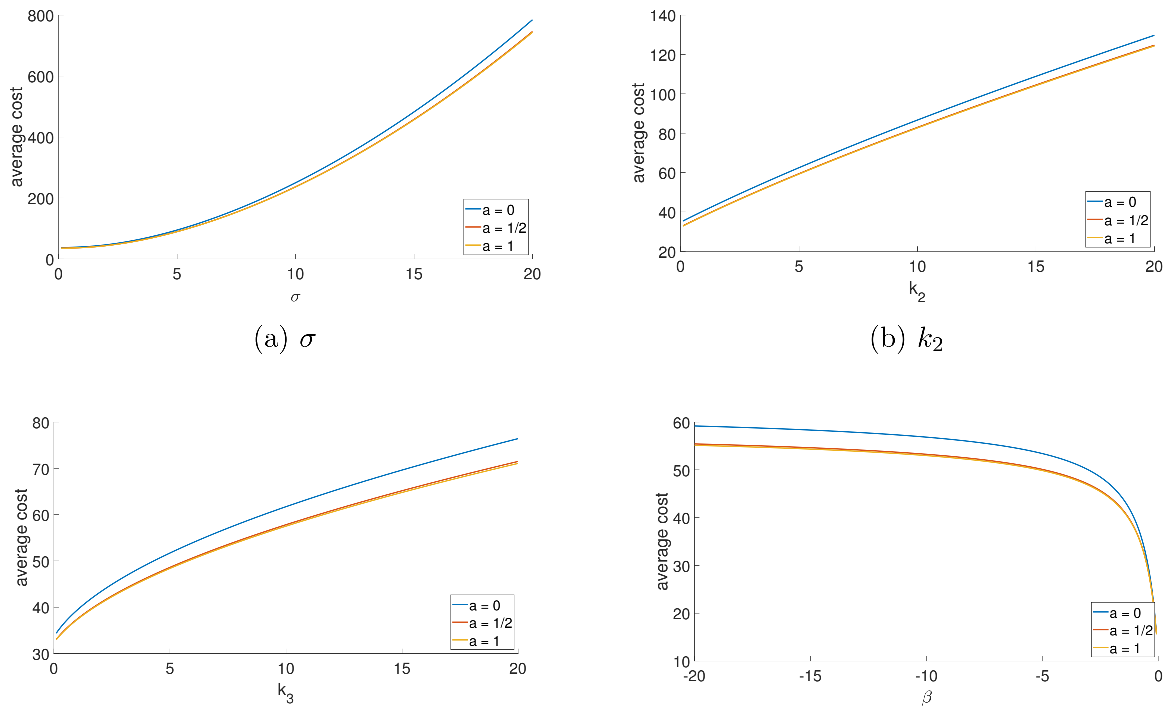
(a) lower threshold

(b) upper threshold

(c) amount of order

(d) average cost

Figure 3.17.: a-dependence in a geometric Brownian motion inventory model with uniformly distributed supply ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ k_3 = 3,\ k_4 = 4$ and $\beta = -2$)



(a) σ

(b) k_2

(c) k_3

(d) β

Figure 3.18.: Sensitivity analysis of the average cost in a geometric Brownian motion inventory model with uniformly distributed supply for different model parameters ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ k_3 = 3,\ k_4 = 4$ and $\beta = -2$, if not variable)

threshold - lower threshold) and the average cost, dependent on a. While the lower threshold nearly stays constant the upper threshold reduces by half between $a = 0$ and $a = 1$. This is also the reason that the amount of order is nearly the same as the upper threshold. The average cost decreases for increasing a, but on a small level. This can be explained by the fact the ordering costs is concave and so the advantage of smaller holding costs is partly offset by smaller savings due to economics of scale for the smaller supplies.

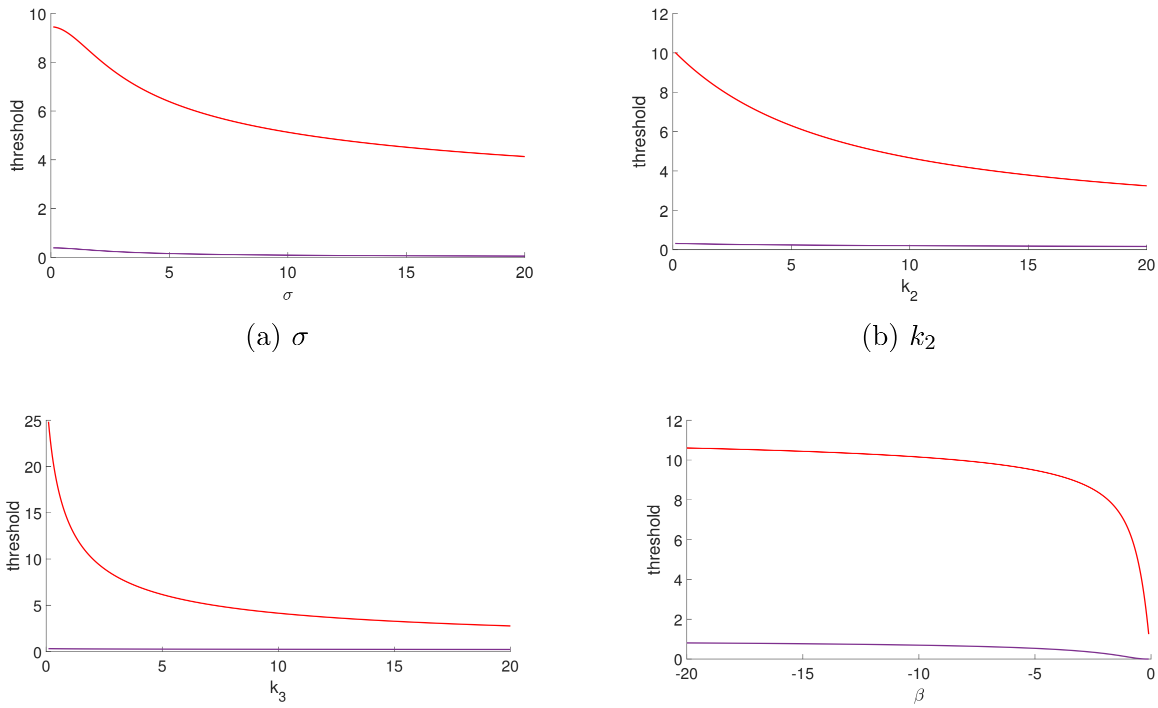
In Figure 3.18 we see the average costs for three different a, depending on σ (cf. Figure 3.18(a)), k_2 , k_3 (cf. Figure 3.18(c)) and β (cf. Figure 3.18(d)), which shows again that there are only very small cost differences between $a = 0.5$ and $a = 1$. For increasing σ there is an exponential cost increase, since the variance for a geometric Brownian motion is $c(\exp(\sigma^2 t) - 1)$ and with increasing variance on the one hand more orders are necessary and on the other hand the inventory reaches higher levels with higher probability and so

the holding costs increasing as well. Figure 3.19 shows the lower and upper thresholds for $a = 0.5$ depending on the same parameters as above. For σ (cf. Figure 3.19(a)) the lower threshold decreases from 0.4 to 0.04 between $\sigma = 0$ and $\sigma = 20$, in the same interval the upper threshold is reduced by half from 9.5 to 4. The reason for this is that for small σ the inventory will nearly be deterministic, while for greater σ the fluctuation is much higher and the smaller upper threshold prevents high holding costs for high inventory levels.

For increasing k_2 we see a linear increase in the average cost (cf. Figure 3.18(b)) , while the lower threshold (cf. Figure 3.19(b)) decreases on very a very low level and the upper threshold decreases from around 10 at $k_2 = 0$ to around 3 at $k_2 = 20$. The reason is that for small k_2 big orders are relative cheap and main cost factor for orders is the fixed order price k_1, so it is less expensive to make fewer large orders, however for large k_2 the main cost factor per order is the order price per unit k_2, so it is cheaper to make more small orders, especially since for large orders the holding costs increase as well.

In case of k_3 there is strong increase in the average cost (cf. Figure 3.18(c)), which flattens out a little bit for increasing k_3. The lower threshold (cf. Figure 3.19(c)) stays nearly constant on a low level, while the upper threshold decreases very quickly for small k_3 and then flattens out from $k_3 = 4$. For small k_3 the holding cost for large inventory is very inexpensive but this advantage decreases very quickly (with rate $\frac{1}{k_3}$) and so the cost increases rapidly close to zero. The larger k_3 gets the more worthwhile it is to have always a small inventory and more frequent orders (the expected order size for $k_3 = 20$ is only around 2), so the increase in the average price is more driven by the more frequent orders.

In contrast, for β there is only a very small decrease for small β before the average cost (cf. Figure 3.18(d)) drops with with a high slope beginning at around $\beta = -2$. Just the same holds for the thresholds (cf. Figure 3.19(d)), which stay nearly on a constant level until $\beta = -3$, where they both drop quickly, the lower threshold is very close to 0 then, while the upper threshold drops from 10 to 1.25. This can be easily explained, since for very small β the holding costs close to zero nearly vanish, so the only relevant costs, in this case are



(a) σ (b) k_2

(c) k_3 (d) β

Figure 3.19.: Sensitivity analysis of the thresholds in a geometric Brownian motion inventory model with uniformly distributed supply for different model parameters ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ k_3 = 3,\ k_4 = 4, \beta = -2,\ p = \frac{1}{2}$, if not variable)

the ordering costs. However for smaller β it is very expensive to have small inventories, so the lower threshold is around 0.9, while the upper threshold is very large, since the holding costs for a large inventory in this case is much less expensive than for small inventory (the expected inventory level after ordering is around 8.2).

3.3.2. Binomial distribution

Let

$$\mathcal{Q}\left(\left\{\frac{(n-k)y}{n+1}+\frac{(k+1)z}{n+1}\right\},y,z\right)=\binom{n}{k}p^k(1-p)^{n-k}=:p_k,\quad k\in\{0,\dots,n\},$$
$$x_{k,y,z}=\frac{(n-k)y}{n+1}+\frac{(k+1)z}{n+1},$$

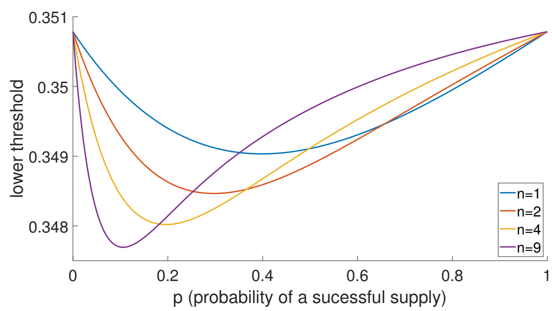
where $p\in[0,1]$ and $n\in\mathbb{N}$ are given parameters. The cost function is as in Subsection 3.1.2 given by

$$J(\tau,Y)=\frac{\sum_{k=0}^{n}\left(p_k(c_1(x_{k,y,z},z)+p_k g_0(x_{k,y,z})\right)-g_0(y)}{\sum_{k=0}^{n}p_k\zeta(x_{k,y,z})-\zeta(y)}.$$

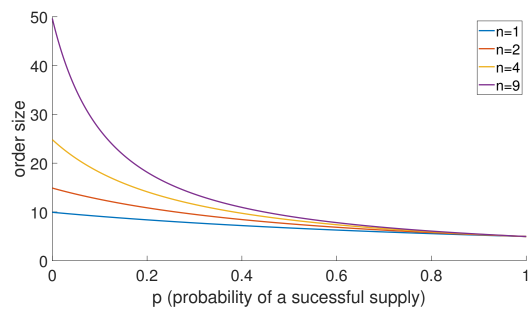
Furthermore let

$$\mu=3,\ \sigma=1,\ k_1=8,\ k_2=2,\ k_3=3\text{ and }k_4=4.$$

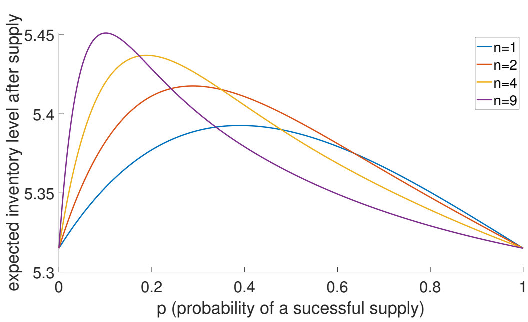
In Figure 3.20(d) we see that for this model the average costs only vary very little (the cost maximum for $n=9$ is only 3% higher than in the deterministic cases ($p=0$, resp. $p=1$). On this small deviation level, however, the average cost structure is very similar to the one of the drifted Brownian motion with binomial supply of subsection 3.1.2. First the average cost curves increase close to $p=0$, which is steeper for greater n, before reaching the maximum between $p=0.05$ and $p=0.5$ (the smaller n the greater is p, for which the maximum is reached). Most of the behavior of the average cost curve can be explained as for the drifted Brownian motion with binomial supply, even though in this model the lower threshold (cf.



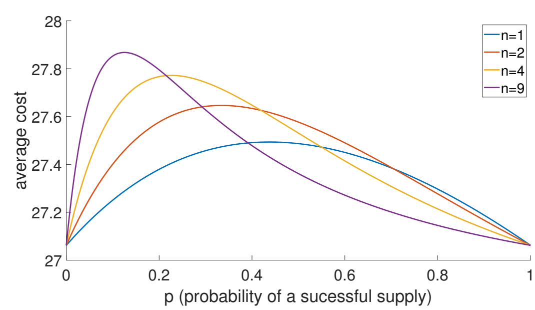
(a) lower threshold



(b) amount of order

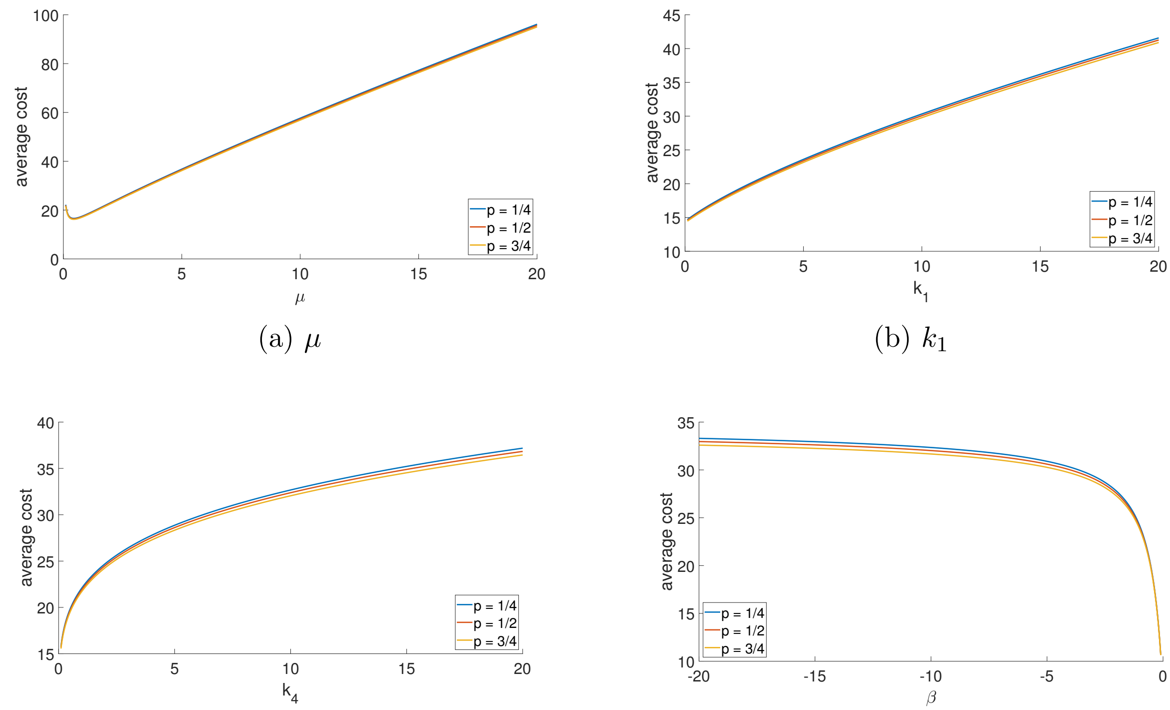


(c) expected inventory level after supply



(d) average cost

Figure 3.20.: p-dependence in a geometric Brownian motion inventory model with binomial distributed supply ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ k_3 = 3,\ k_4 = 4$ and $\beta = -2$)



(a) μ (b) k_1 (c) k_4 (d) β

Figure 3.21.: Sensitivity analysis of the average cost in a geometric Brownian motion inventory model with binomial distributed supply for different model parameters ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ k_3 = 3,\ k_4 = 4$ and $\beta = -2$, if not variable)

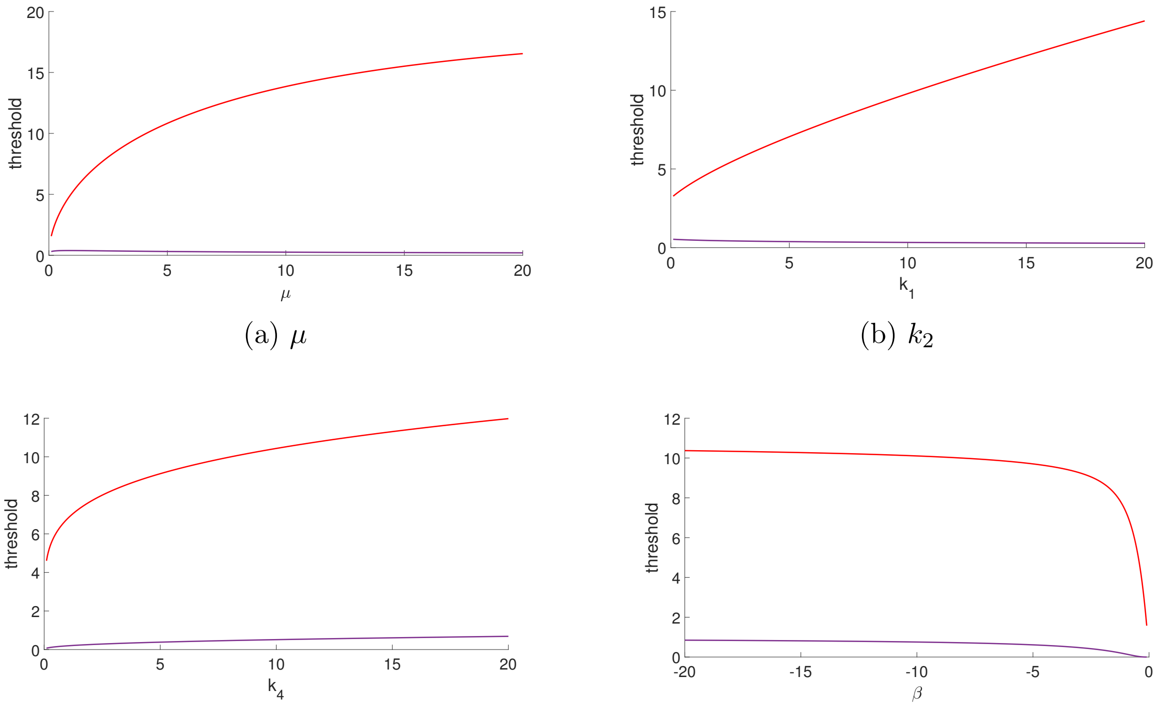
Figure 3.20(a) has it maximum at $p = 0$, resp. $p = 1$ (however the maximum deviation is only around 0.1%). The expected inventory level after supply curves (cf. Figure 3.20(c) are very similar to the average cost curves (again with very little deviation). This means in this case the order size is chosen in a way that the risk of small supplies and therefore more orders is relatively small. Therefore the risk of high supplies with associated higher holding costs increases. This is a consequence of the possible economics of scale, so larger orders are tend to be less expensive. Meaning in the case of uncertain supply it is cheaper to make too large orders with corresponding large holding costs, while at least saving some money for the large order, than making a lot of small orders.

In Figure 3.21 we see the average cost for $n = 4$ and different p ($p = 0.25$, $p = 0.5$ and $p = 0.75$) depending on σ, k_2, k_3 and β. As we already saw above the difference in the average costs for different p is very small with $p = 0.75$ having the cheapest average costs,

while $p = 0.25$ is the most expensive for the three p. We further note that the average cost curve as well as the threshold curves (cf. Figure 3.22) look very similar to the ones of the uniformly distributed random supply (Figure 3.18 & Figure 3.19). This also hold true for all other parameters, so for the sake of avoiding repetition we refer to subsection 3.3.1 for an analysis of these parameters and focus here on the other three parameter of the model. For μ we see similar to the drifted Brownian motion model a decrease in the average cost (cf. Figure 3.21(a) close to zero reaching a minimum at around $p = 0.4$ and afterwards increasing linearly. For $\mu = 0$ again our conditions would be violated and as before in subsection 3.1.1 the average cost tend to infinity for $\mu \to 0$. In this case both thresholds (cf. Figure 3.22(a)) converge to zero, which leads to very large order costs. For increasing μ the upper threshold increases, since in for greater μ the order frequency (for constant threshold) increases. However the increase flattens out for larger μ to avoid great holding cost, so the increase in the average cost is driven by both increase in the order cost (through more frequent orders) and the increase in holding costs.

For k_1 the average costs (cf. Figure 3.21(b) increase nearly linearly, but not as steeply as for μ, while the upper threshold (cf. Figure 3.22(b)) also increases almost linearly (only close to $k_1 = 0$ is the increase a bit steeper) and the lower threshold decreases slowly. For small k_1 it is very inexpensive to order (the only important cost in this case is the order cost per unit) so it makes sense to order often to save money on the holding costs. When, however k_1 gets larger each order has high fix costs, so it is cheaper to pay higher holding costs but therefore order less frequently.

In the case of k_4 there is a steep increase in the average cost (cf. Figure 3.21(c)) for k_4 close to zero, which flattens the larger k_4 becomes. The lower threshold (cf. Figure 3.22(c)) increases from nearly zero for k_4 near to zero to 0.7 for $k_4 = 20$, while the upper threshold increases relatively quickly close for small k_4 and flattens a bit for increasing k_4. This can be explained by the fact that for $k_4 \approx 0$ the holding costs for inventory levels close to zero are very small and therefore it makes sense to delay the orders until a very small inventory level



(a) μ (b) k_2

(c) k_3 (d) β

Figure 3.22.: Sensitivity analysis of the thresholds in a geometric Brownian motion inventory model with binomial distributed supply for different model parameters ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ k_3 = 3,\ k_4 = 4$ and $\beta = -2$, if not variable)

is reached and only make relative small orders. On the other side for great k_4 the holding cost for small inventory levels are very expensive and so the goal is it to avoid such small inventory levels by increasing the point of order and the amount of order.

3.3.3. Truncated normal distribution

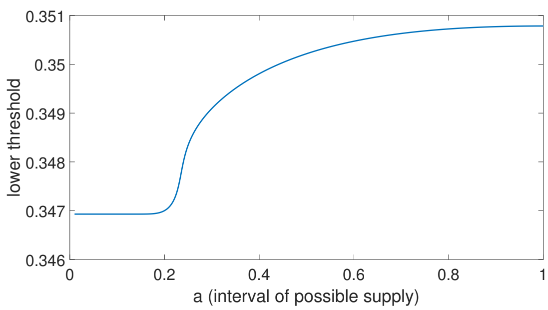
Let $\tilde{X}$ a normal distributed random variable with mean $\tilde{\mu}$ and standard deviation $\tilde{\sigma}$ (it is important to be careful to not confuse $\tilde{\mu}$ and $\tilde{\sigma}$ with the parameters μ and σ of the inventory process). We can't use this as a possible supply distribution, since it is not bounded. However, if we define $X = \tilde{X}|\tilde{X} \in [a, b]$ we get the so called truncated normal distribution with the density

$$\begin{aligned} f(x) &= \frac{\phi\left(\frac{x-\mu}{\sigma}\right)}{\sigma\left(\Phi\left(\frac{b-\mu}{\sigma}\right) - \Phi\left(\frac{a-\mu}{\sigma}\right)\right)} \\ \text{with } \phi(x) &= \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}x^2\right) \\ \text{and } \Phi(x) &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} \exp\left(-\frac{1}{2}t^2\right) \mathrm{d}t. \end{aligned}$$

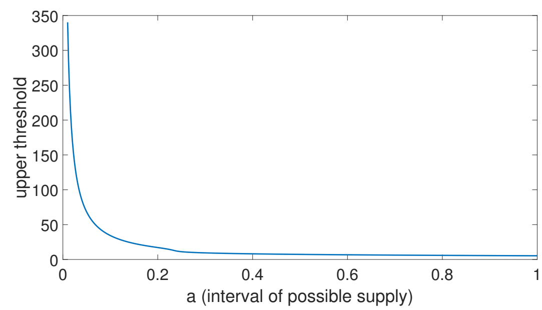
In this subsection we will assume that the supply is truncated normal distributed on the interval $[p_a, z]$, where $p_a = (1-a)y + az$ is defined as before. This kind of distribution makes it impossible to compute simple antiderivatives for c_1, g_0 and ζ. Nevertheless we can analyze this model. Therefore we set

$$\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ k_3 = 3,\ k_4 = 4, \beta = -2, \tilde{\mu} = 2,\ \tilde{\sigma} = 4.$$

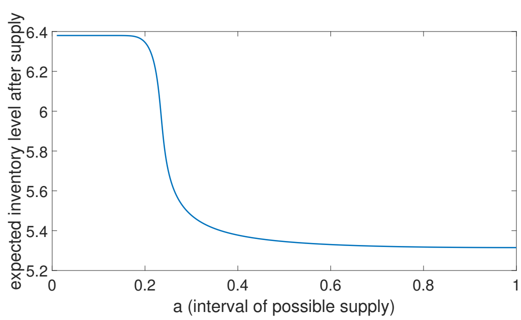
In Figure 3.23(d) we can see surprising results for the cost structure in this model. The average cost is nearly constant between $a = 0$ and $a \approx 0.25$. For $a > 0.25$ the average cost decreases more quickly, but still the overall deviation of the cost is relatively small (only around 2%). A very similar behavior can be seen for the lower threshold (cf. Figure 3.23(a)), which is constant until $a = 0.25$. With increasing a the lower threshold increases



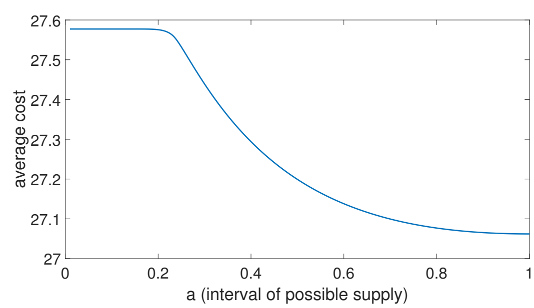
(a) lower threshold



(b) upper threshold

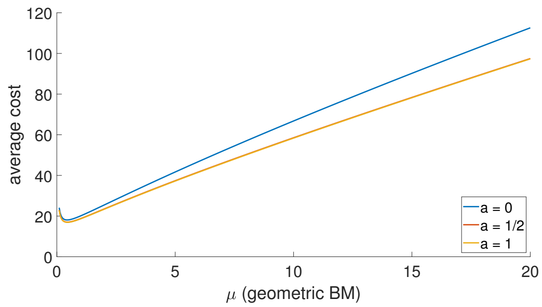


(c) expected inventory level after supply

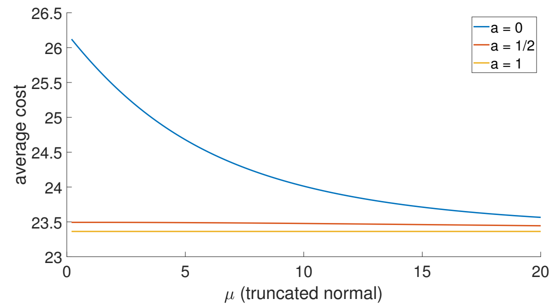


(d) average cost

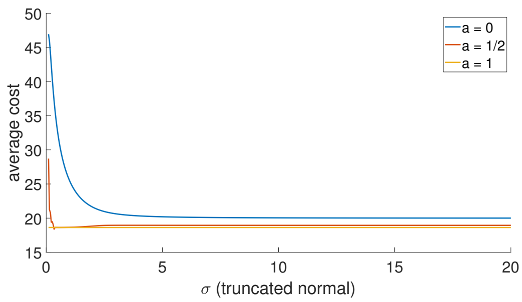
Figure 3.23.: a-dependence in geometric Brownian motion inventory model with truncated normally distributed supply depending ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ k_3 = 3,\ k_4 = 4,\ \beta = -2, \tilde{\mu} = 2$ and $\tilde{\sigma} = 4$)



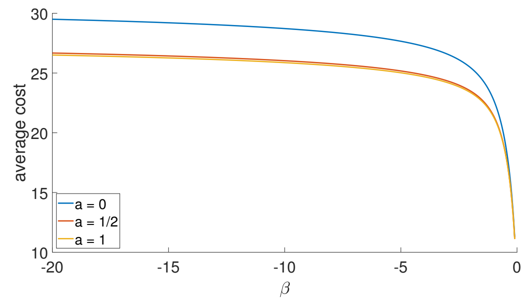
(a) μ (geometric Brownian motion)



(b) $\tilde{\mu}$ (truncated normal distribution)



(c) $\tilde{\sigma}$ (truncated normal distribution)



(d) β

Figure 3.24.: Sensitivity analysis of the average cost in a geometric Brownian motion inventory model with truncated normal distributed supply for different model parameters ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ k_3 = 3,\ k_4 = 4, \beta = -2,\ \tilde{\mu} = 2,\ \tilde{\sigma} = 4$, if not variable)

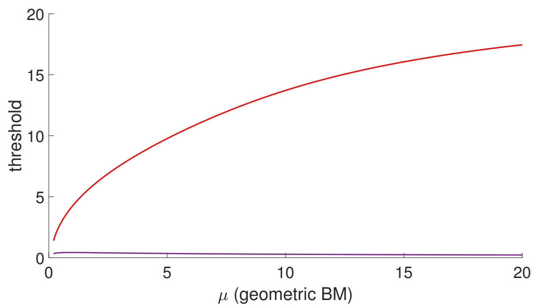
very slowly. A very different curve can be seen for the upper threshold (cf. Figure 3.23(d)), which decrease very quickly for $a < 0.25$ and then flattens out. This very high values for the upper threshold are needed to hold the the expected inventory level after supply (cf. Figure 3.23(c)) constant (the expected inventory level after supply is given by $\tilde{\mu} + \frac{\phi(p_a)-\phi(z)}{\Phi(z)-\Phi(p_a)}\tilde{\sigma}$). This model behaves very counterintuitively and is therefore difficult to explain. The high values for the upper threshold are a consequence of the lower probability mass for values close to z for small values of a. There are close to no extra costs arising from the risk of high supplies in this case.

The next step is a sensitivity analysis of different model parameters (μ, $\tilde{\mu}$, $\tilde{\sigma}$ and β of the average cost for three different a (cf. Figure 3.24) and the thresholds for $a = \frac{1}{2}$ (cf. Figure 3.25). For μ (cf. Figure 3.24(a) & Figure 3.25(a)) and β (cf. Figure 3.24(b) & Figure

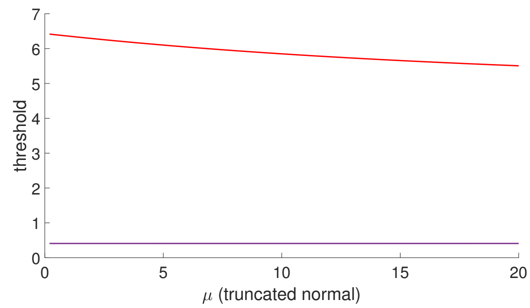
3.25(b)) we can see the same behavior as in the subsections before (cf. Figure 3.21 & Figure 3.22). Therefore we dispense on the analysis of these two parameters and instead take a closer look on $\tilde{\mu}$ and $\tilde{\sigma}$ (the parameters of the truncated normal distribution).

In Figure 3.24(b) we see that the average cost are decreasing with increasing $\tilde{\mu}$ especially for $a = 0$. For $a = \frac{1}{2}$ and $a = 1$ the average cost are almost constant. This result again is as expected, since with increasing $\tilde{\mu}$ the probability of getting a supply close to the upper threshold increases. However it is a little bit surprising that for $a = \frac{1}{2}$ and $a = 0$ the average cost stays constant. In Figure 3.25(b) we see that the thresholds stay almost constant in this case as well.

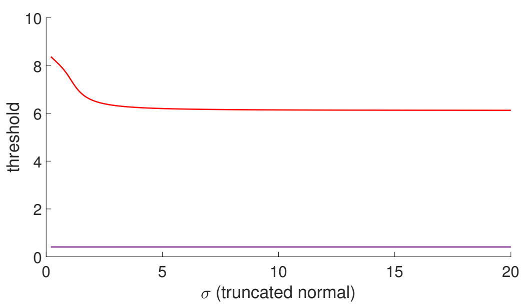
For $\tilde{\sigma}$ the average cost curve (cf. Figure 3.24(c)) looks similar to the one of the $\tilde{\mu}$. However the decrease for $a = 0$ is much steeper in this case. Also there is in contrast to $\tilde{\mu}$ for $a = \frac{1}{2}$ a steeps decrease close to $\tilde{\sigma} = 0$. For $\tilde{\sigma} > 2$ the average cost stays almost constant. Again the reason is the with a truncated normal distribution the probability for getting a supply close to the upper threshold increases. Nevertheless this is surprising, since a higher variance normally leads to higher costs. The lower threshold (cf. Figure 3.25(c)) is almost constant expect from a small decrease of the upper threshold between $\tilde{\sigma} = 0$ and $\tilde{\sigma} = 2$. For the upper threshold we see a decrease between $\tilde{\sigma} \in [0, 2]$, while for $\tilde{\sigma} > 2$ the upper threshold stays almost constant.



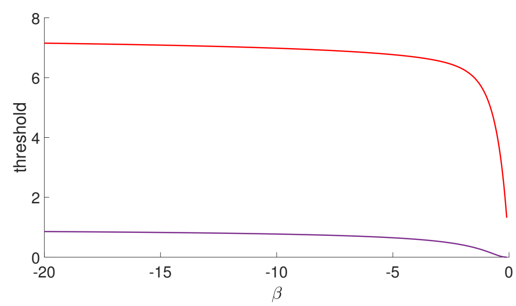
(a) μ (geometric Brownian motion)



(b) $\tilde{\mu}$ (truncated normal distribution)



(c) $\tilde{\sigma}$ (truncated normal distribution)



(d) β

Figure 3.25.: Sensitivity analysis of the thresholds in a geometric Brownian motion inventory model with truncated normal distributed supply for different model parameters ($\mu = 3,\ \sigma = 1,\ k_1 = 8,\ k_2 = 2,\ k_3 = 3,\ k_4 = 4, \beta = -2,\ \tilde{\mu} = 2,\ \tilde{\sigma} = 4$, if not variable)

4. Summary

At first glance the problem we tried to solve looked very complicated. Not only did we have a random inventory level based on the stochastic differential equation, but also the order supply is random. In this difficult-looking model our goal was to find a optimal order strategy. However, as explained in the first chapter, this problem can be reformulated as non-linear optimization problem. Even if the functions we need to optimize are very complicated and partly contain integrals without explicit antiderivative, with the help of the SQP-algorithm and the Gauss-Kronrod-algorithm we were able to solve the problem.

This thesis showed that this method not only work for a wide range of different models. On the one hand it can handle different inventory models with some of them allowing negative inventory levels and some allowing this not. Also the models had different cost structures with differences in having economics of scale and dealing with very small inventory levels, which can have bounded or possibly unbounded cost. On the other hand we tested the method on a wide variety of different supply distributions. We showed that it not only works for continuous distributions but also for discrete distributions as the binomial distribution. We also tested this for some easier distributions as the uniform distribution, but also for more complex one as the beta distribution, the truncated normal distribution and compound polynomial distribution. In this case we also were in the need of numerical integration algorithms, since some of the integrals we need did not have an (simple) antiderivate.

Many of the results we got were very intuitive, so that this thesis confirmed what one would expect. However some results are unexpected and are more difficult to explain. We saw

that the average cost in a different models for μ (the drift parameter of the inventory level) close to zero is very high, which was very surprising at least at the first glance. For the truncated normal distributed supply we saw that in a geometric Brownian motion model the average cost is almost constant for $a \in [0, \frac{1}{4}]$. Another interesting result this thesis showed, were the interesting curves for the thresholds in a drifted Brownian motion with reflection at zero inventory model, where the supply is polynomial distributed. Here we saw interesting peak in the upper threshold, where the curve is not differentiable. However the curve of the average cost looks still smooth.

Despite lacking a formal proof of a convergence result, this thesis showed on a experimental basis that the SQP algorithm is a very useful tool for finding the optimal order points. It can be assumed that this method would also work for different models with other distributions for the supply.

5. Bibliography

[1] Chung, K.L. and Williams, R.J. (1983). *Introduction to Stochastic Integration* Birkhäuser, Boston

[2] Helmes K.L., Stockbridge, R.H. and Zhu, C. (2018). *A weak convergence Approach to inventory control using a long-term average criterion.* Advances Applied Probability **50** 1032-1074.

[3] Helmes K.L., Stockbridge, R.H. and Zhu, C. (2017). *Continuous inventory models of diffusion type: Long term average cost criterion.* The Annual of Applied Probability 2017, Vol. 27, No. 3, 1831-1885.

[4] Helmes K.L., Stockbridge, R.H. and Zhu, C. *Single-Item Continuous-Review Inventory Models with Random Supplies.* article in preparation

[5] Krommer, A.R., Ueberhuber, C.W. (1998). *Computational Integration.* Siam, Philadelphia.

[6] Nocedal, J. and Wrigth, S.J (1999). *Numerical Optimization.* Springer, New York.

[7] Notaris, S.E. (2016). *Gauss-Kronrod-quadrature formulae - A Survey of fifty years of research.* Electronic Transactions on Numerical Analysis Volume **45** 371-404

[8] Smith H.V. (1993). *Numerical Methods of Integration.* Chartwell-Bratt, Bromley.

A. Numerical algorithms for analyzing the models

In the next chapter we will analyze different models and especially try to find the (y, z)-policy (we will call it lower threshold and upper threshold), which minimizes the costs for different models. However the functions we try to minimize are too complex to find closed form solutions and we have to pay attention to various constraints which apply. Therefore we need a powerful algorithm to solve these optimization problems. For this thesis we chose sequential quadratic programming algorithms, since they tested as most stable. Besides that some of our models contain integrals, which can not be solved explicitly, in these cases we also need numerical algorithms to solve these integrals. MATLAB, which is used for all computations of this thesis uses a global quadrature, similar to the Gauss-Kronrod-quadrature. However the development and implementation of such numerical algorithms is a very wide and complicated field, so we can only give a brief overview at these algorithms. The goal here is to give one basic algorithm for each problem. In practice there are more advanced algorithms with better theoretical properties.

Numerical optimization: Sequential Quadratic Programming

In this section we will develop with the help of [6] the idea of a local SQP algorithm. Let us start with the following problem:

$$\begin{aligned} &\min \ f(x) \\ \text{subject to } &b_i(x) \geq 0, \quad i \in \mathcal{I} \\ &c_i(x) = 0, \quad i \in \mathcal{E}. \end{aligned} \tag{A.1}$$

We define the Lagrangian for this problem as

$$\mathcal{L}(x, \lambda) = f(x) - \sum_{i \in \mathcal{I}} \lambda_i b_i(x) - \sum_{i \in \mathcal{E}} \lambda_i c_i(x), \tag{A.2}$$

$\mathcal{A}(x) = \mathcal{E} \cup \{i \in \mathcal{I} : b_i(x) = 0\}$ as the active set and the active set $\mathcal{A}(x^*)$ of the optimal solution x^* as optimal active set. We say a point x' fulfills the linear independence constraint qualification (LICQ) if the set of all active constraint gradients $\{\nabla c_i(x'), i \in \mathcal{A}(x')\}$ are linearly independent.

Theorem A.1 (Karush-Kuhn-Tucker conditions)
Suppose x^* is a local solution for (A.1). Then there is a vector λ^* (Lagrange multiplier vector), such that the following conditions are satisfied:

$$\nabla_x \mathcal{L}(x^*, \lambda^*) = 0 \tag{A.3a}$$

$$c_i(x^*) = 0, \ \forall i \in \mathcal{E} \tag{A.3b}$$

$$b_i(x^*) \geq 0, \ \forall i \in \mathcal{I} \tag{A.3c}$$

$$\lambda^* c_i(x^*) = 0, \ \forall i \in \mathcal{E} \cup \mathcal{I} \tag{A.3d}$$

$$\lambda^* \geq 0, \ \forall i \in \mathcal{I} \tag{A.3e}$$

We note that in the case of having no inequality constraints by denoting the Jacobian matrix of the constraints as

$$A(x)^T = (\nabla c_1(x), \cdots, \nabla c_m(x)),$$

we need to solve the system

$$F(x, \lambda) = \begin{pmatrix} \nabla f(x) - A(x)^T \lambda \\ c(x) \end{pmatrix} = 0.$$

This system can be approximately solved using Newton's method by finding the solution of the KKT system

$$\begin{pmatrix} W_k & -A_k^T \\ A_k & 0 \end{pmatrix} \begin{pmatrix} p_k \\ p_\lambda \end{pmatrix} = \begin{pmatrix} -\nabla f_k + A_k^T \lambda_k \\ -c_k \end{pmatrix} \tag{A.4}$$

and iterating

$$x_{k+1} = x_k + p_k$$

$$\lambda_{k+1} = \lambda_k + p_\lambda$$

with W_k the Hessian matrix of the Lagrangian ($W_k = \nabla^2_{xx} \mathcal{L}(x_k, \lambda_k)$).
The basic idea of the SQP-algorithm is to linearize the function as well as the constraints and solve these problems iteratively. So this leads to the following problem:

$$\min \nabla f_k^T p + \frac{1}{2} p^T W_k p \tag{A.5a}$$

$$\text{subject to } c_i(x_k) + \nabla c_i^T(x_k) p = 0, \ \forall i \in \mathcal{E} \tag{A.5b}$$

$$b_i(x_k) + \nabla b_i^T(x_k) p \geq 0, \forall i \in \mathcal{I}. \tag{A.5c}$$

There is no need to consider the term $f(x_k)$ in (A.5a), since this term is constant. If we

suppose we only have equality constraints this optimization problem can be solved locally with the Newton method. Using the KKT-conditions we get that the unique solution (p_k, μ_k) satisfies

$$\begin{aligned} W_k p_k + \nabla f_k - A_k^T \mu_k &= 0 \\ A_k p_k + c_k &= 0, \end{aligned}$$

so by using $(A.4)$ we get

$$\begin{pmatrix} W_k & -A_k^T \\ A_k & 0 \end{pmatrix} \begin{pmatrix} p_k \\ \lambda_{k+1} \end{pmatrix} = \begin{pmatrix} -\nabla f_k \\ -c_k \end{pmatrix}.$$

However if there are also inequality constraints we need to use different methods as active set algorithms, which solves problems such as $(A.5)$. We will assume that the linearized inequalities are not unsolvable, which can happen as we see on the following example with the inequalities $x^2 \geq 2$ and $x \leq 1$ (with the solution $x \in (-\infty, -\sqrt{2})$). However, if we use a Taylor expansion of degree 2 at $y = 3$ to linearize these inequalities we get $3 + x \leq 1$ and $9 + 6x \geq 0$, which can be simplified to $x \leq -3$ and $x \geq -\frac{16}{9}$. This system obviously does not have a solution anymore. One can show that in this case there exists an equivalent problem with the same solution (for γ large enough and e^T the unit vector).

$$\begin{aligned} &\min f(x) + \gamma e^T(v + w) \\ \text{subject to } &c_i(x) - v_i + w_i = 0 \ i \in \mathcal{E} \\ &b_i(x) - v_i + w_i \geq 0 \ i \in \mathcal{I} \\ &v \geq 0 \\ &w \geq 0. \end{aligned}$$

First we note that we can write the KKT-conditions from Theorem $A.1$ as

$$W_k x^* + \nabla f_k - \sum_{i \in \mathcal{A}(x^*)} \lambda^* = 0 \tag{A.6a}$$

$$a_i^T x^* = b_i \ \forall i \in \mathcal{A}(x^*) \tag{A.6b}$$

$$a_i^T x^* \geq b_i \ \forall i \in \mathcal{I} \setminus \mathcal{A}(x^*) \tag{A.6c}$$

$$\lambda_i^* \geq 0 \ \forall i \in \mathcal{I} \cap \mathcal{A}(x^*), \tag{A.6d}$$

where we write for simplicity $a_i^T p = b_i$ for the corresponding constraint in $(A.5b)$. We assume that W_k is positive semidefinite. The algorithm starts with a feasible x_0. We define the working set $\mathcal{W}_j$ for the jth iterate x_j as a subset of the active set $\mathcal{A}(x_j)$, which includes $\mathcal{E}$ and and some (possibly all) active inequality constraints, such that all gradients of the constraints in the working set are linearly independent. For all iterates x_j we verify if x_j is the minimal solution of our quadratic problem on the working set. If this is not the case we solve the quadratic problem, where we only use the constraints of the working set $\mathcal{W}_j$ and make the inequality constraints to equality constraints, so we try to solve

$$\min_p \frac{1}{2} p^T W_k p + \nabla f_k^T p \tag{A.7a}$$

$$\text{subject to } a_i^T p = b_i \ \forall i \in \mathcal{W}_j, \tag{A.7b}$$

This can be solved with the Newton method explained above and leads to the solution p_j. All constraints active at x_j are still active at $x_j + \alpha p_j$, since $a_i^T(x_j + \alpha p_j) = a_i^T x_j$ for all values of α. We now want to move along the direction p_j, such that all constraints are still satisfied, so if possible we define $x_{j+1} = x_j + p_j$. If however some constraints are not fulfilled at this new point we have to the find the maximum α_j, such that for $x_{j+1} = x_j + \alpha_j p_j$ all constraints are satisfied. There is an explicit formula for α_j, since we only need to look at

the constraints $i \notin W_j$ with $a_i^T p_j < 0$, where we have $a_i^T(x_j + \alpha_j p_j)$ if and only if

$$\alpha_j \leq \frac{b_i - a_i^T x_j}{a_i^T p_j},$$

so we can define

$$\alpha_j = \min\left(1, \min_{i \notin \mathcal{W}_k, a_i^T p_j < 0} \frac{b_i - a_i^T x_j}{a_i^T p_j}\right).$$

The corresponding constraint for which the minimum above is achieved is called the blocking constraint (it is possible that no blocking constraint exists in the case $\alpha_j = 1$ and no new constraint becomes active). The working set W_{j+1} is constructed by adding the blocking constraint. We continue as long as we find a point $\hat{x}$, that minimizes the quadratic problem on the latest working set $\hat{\mathcal{W}}$. In this iteration step $p = 0$ and so we can compute $\hat{\lambda}_i$ as solution of the system

$$\sum_{i \in \hat{W}} a_i \hat{\lambda}_i = W_k \hat{x} + \nabla f_k \tag{A.8}$$

and so $(A.6a) - (A.6c)$ are fulfilled at this point. If additionally all Langrange multiplier of the inequality constraints in the working set are nonnegative, then $(A.6d)$ is fulfilled as well and $\hat{x}$ is a local minimizer. But if there is at least one negative multiplier $\hat{\lambda}_i$, $i \in \hat{\mathcal{W}} \cap \mathcal{I}$, then $(A.6d)$ is not satisfied and so it can be possible that the function can be decreased even more by dropping this constraint. In practice the constraint is dropped for which the multiplier $\hat{\lambda}_i$ takes the smallest value. One can show that this leads to a descent direction of the objective function in the next step. So this leads to the following algorithm.

Algorithm A.2 (Local SQP algorithm using an active set method)

Calculate a starting point x_0, which is feasible

Define $\mathcal{W}_0$ as subset of active constraints at x_0 (gradients are linearly idependent)

while x_j is not the minimal solution

Use Newton's method to solve $(A.7a)$ and find p_j

if $p_j = 0$

Solve $(A.8)$ to get the Lagrange multiplier $\hat{\lambda}_i$

if $\hat{\lambda}_i < 0$ for at least one $i \in \mathcal{W}_j \cap \mathcal{I}$

$l = \arg\min_{l \in \mathcal{W}_k \cap \mathcal{I}} \hat{\lambda}_i$

$x_{j+1} = x_j$, $\mathcal{W}_{j+1} = \mathcal{W}_j \setminus \{l\}$

else

solution: $x^* = x_j$ **break**

else

$\alpha_j = \min\left(1, \min_{i \notin \mathcal{W}_k, a_i^T p_j < 0} \frac{b_i - a_i^T x_j}{a_i^T p_j}\right)$

$x_{j+1} = x_j + \alpha_j p_j$

if $\exists$ blocking constraints

$\mathcal{W}_{j+1} = \mathcal{W}_j \cup \{\text{one of the blocking constraints}\}$

else

$\mathcal{W}_{j+1} = \mathcal{W}_j$

end while

This algorithm however may not work in remote starting points of the solution, since it is possible that in such points the Hessian matrix W_k is not positive definite. In this case in practice often approximations of the Hessian such as Full Quasi-Newton and reduced Hessian approximations are used. For fast convergence from remote starting points most practical implementations do use line search or trust region algorithms with approximations of the Hessian, which is often very hard to compute in practice. However these algorithms are very advanced and would go beyond the scope of this thesis.

Numerical integration: Gauss-Kronrod-Quadrature

For many functions, some of them we need in our analysis there does not exist an analytic antiderivative or it is very hard to compute it. In such cases we use numerical algorithms to solve integrals of the corresponding functions. In this subsection we will illustrate an often used approach the Gauss-Kronrod-quadrature, an adaptive method. We will follow [5] and [7] closely. A quadrature formula is an approximation of the integral

$$\int_a^b w(x) f(x) \mathrm{d}x,$$

with $w(x)$ a given weight-function. We assume that f is well-behaved (in our analysis all functions are continuous, which is sufficient). The approximations are given by

$$\int_a^b w(x) f(x) \mathrm{d}x = \sum_{i=1}^{n} w_i f(x_i) + \varepsilon, \tag{A.9}$$

where ε is the error, which should be close to zero (or even zero for some classes of functions). There are a lot of different possibilities for choosing the w_i and $x_i \in (a, b)$, which results in different quadrature formulas. One in comparison simple method is the Newton-Cotes-quadrature, which chooses the x_i to be equidistant in (a, b). This method is exact for all $p \in \Pi_{n-1}$ (polynomials with degree smaller or equal $n-1$, with n the number of abscissas). The weights w_i are calculated based on the principle of interpolation, meaning solving the

system of linear equation

$$\sum_{i=1}^{n} w_i = \int_a^b w(x)\mathrm{d}x$$
$$\sum_{i=1}^{n} w_i x_i = \int_a^b x w(x)\mathrm{d}x$$
$$\vdots$$
$$\sum_{i=1}^{n} w_i x_i^{n-1} = \int_a^b x^{n-1} w(x)\mathrm{d}x,$$

where the integrals on the right side should be solved analytically. The system can be solved uniquely, since the coefficient matrix is the Vandermonde-matrix V, which has determinant

$$\det V = \prod_{i=2}^{n}\prod_{j=1}^{i-1}(x_i - x_j) \neq 0,$$

for $x_i \neq x_j \; \forall i \neq j, i \in \{1, ..., n\}$. A better method in the sense that it solves all polynomials of degree $2n-1$ or smaller is the Gaussian quadrature formula. Here not only the weight can be chosen, but also the abcissas are variable. This is also the maximal degree for which a quadrature formula can be exact. The idea of determining the abscissas is to use the Euclidean algorithm to write an arbitrary polynomial $P \in \Pi_{2n-1}$ as

$$P(x) = t(x)\nu_n(x) + v(x),$$
$$\text{with } \nu_n(x) = \prod_{i=1}^{n}(x - x_i),$$

where $x_1, ..., x_n$ are the unknown abscissas and $t, v \in \Pi_{n-1}$. As a consequence we can write

$$\begin{aligned}\int_a^b w(x)P(x)\mathrm{d}x &= \int_a^b w(x)t(x)\nu_n(x)\mathrm{d}x + \int_a^b w(x)v(x)\mathrm{d}x \\ &= \sum_{i=1}^n w_i t(x_i)\nu_n(x_i) + \sum_{i=1}^n w_i v(x_i) + \varepsilon_P \\ &= \sum_{i=1}^n w_i v(x_i) + \varepsilon_P,\end{aligned}$$

due to the fact that $\nu_n(x_i) = 0 \ \forall i \in \{1, \cdots, n\}$. The goal now is to ensure that $\varepsilon_P = 0$. We already know that we can use the principle of interpolation to find weights $w_1, \cdots, w_n$ to create an exact formula

$$\int_a^b w(x)v(x)\mathrm{d}x = \sum_{i=1}^n w_i v(x_i),$$

so if it is possible to choose the abscissas such that for all $t \in \Pi_{n-1}$ the formula

$$\int_a^b w(x)t(x)\nu_n(x)\mathrm{d}x = 0 \tag{A.10}$$

we would have that $\varepsilon_P = 0$ and since we chose $P \in \Pi_{2n-1}$ arbitrarily this leads to a quadrature formula for all $p \in \Pi_{2n-1}$. To find the right abscissas we need to introduce orthogonal polynomials. For an arbitrary weight function w, we define an inner product on $\pi(a, b)$ as

$$\langle p, q \rangle_w := \int_a^b w(x)p(x)q(x)\mathrm{d}x$$

We say two polynomials p and q are w-orthogonal if $\langle p, q \rangle_w = 0$. Using the Gram-Schmidt-orthogonalization process on the set $\{1, x^2, x^3, \cdots\}$ we can find a set $\mathcal{P}^w := \{p_d^w, d \in \mathbb{N}_0\}$ of pairwise w-orthogonal polynomials with deg $p_d^w = d$. The Theorem of Gram-Schmidt states that polynomials of two different w-orthogonal polynomials differ only by multiplicative constants, which especially means they have the same roots. For the practical calculation

there exist a recursion formula, see [5]. So to fulfill $(A.10)$ we know ν_n has to be w-orthogonal to all polynomials in $\Pi_{n-1}[a,b]$, which means $\nu_n = cp_d^w$ and so $x_1, \cdots, x_n$ are the roots of the w-orthogonal polynomial p_d^w. One can show that these roots are all simple roots in the interval $[a,b]$.

The most important weight-function is $w(x) = 1$ on the interval $[-1,1]$, which is called the Gauss-Legrende integration. As long as both and a and b are finite this integration method can solve integrals with arbitrary start and endpoints a and b by the transformation

$$\int_a^b f(x)\mathrm{d}x = \frac{b-a}{2}\sum_{i=1}^{n} w_i f\left(\frac{b-a}{2}x_i + \frac{a+b}{2}\right) + \varepsilon. \tag{A.11}$$

In this case the w-orthogonal polynomials and weights are given by

$$\begin{aligned} p_n^w(x) &= \frac{1}{2^n n!}\frac{d^n}{dx^n}(x^2-1)^n. \\ w_i &= \frac{2(1-x_i)^2}{n^2(p_{n-1}^w(x_i))^2} \end{aligned} \tag{A.12}$$

The roots $x_1, \cdots, x_n$ and weights for these functions can be found in tables i.e. in [8] and are often already implemented in different computational programs.

However a weakness of this approach is that the values of the function computed in the nth step can not be used for the the calculation of the $(n+1)$th step, since the roots of the nth orthogonal polynomial are different from all roots of all other orthogonal polynomials. This is especially a problem for the error estimation, which is often done by computing the difference of the estimate of the integral of $(n+1)$st and the nth Legendre-polynomial. There also exmaples known, where this method can lead to inaccurate estimations, since the difference between the $(n+1)$st estimation and the nth estimation is very small, nevertheless both estimations are relatively far away from the exact result, see [8]. This problem made the Russian mathematican Kronrod develop a extended algorithm, which uses the roots of the Legendre-polynomials and $n+1$ new abscissas and is exact for all $p \in \Pi_{3n+1}$. This is a large improvement compared to the Gauss quadrature, where one has to compute with

the same numerical effort $n+1$ new function values to only get an exact solution for all $p \in \Pi_{2n+1}$. The corresponding formula is

$$\int_{-1}^{1} f(x)\mathrm{d}x = \sum_{i=1}^{n} w_i^G f(x_i^G) + \sum_{i=1}^{n+1} w_i^K f(x_i^K) + \varepsilon^{GK}, \tag{A.13}$$

where x_i^G are the abscissas of the Gauss-quadrature. However the weights w_i^G are not the same as in the Gauss quadrature, but again can be found in tables. The new abscissas $x_1^K, \cdots, x_n^K$ are the roots of the so-called Stieltjes-polynomials, which are a series of polynomials $(\tilde{p}_n^w)_{n\in\mathbb{N}}$, such that

$$\int_{-1}^{1} \tilde{p}_{n+1}^w(x) p_n^w(x) t^n \mathrm{d}x = 0 \ \forall n \in \mathbb{N}_0.$$

It can be shown that these polynomials are the $\alpha = \frac{1}{2}$-Gegenbauer-polynomials

$$C_n^{\frac{1}{2}}(x) = \frac{1}{\Gamma(\frac{1}{2})} \sum_{i=1}^{\lfloor \frac{n}{2} \rfloor} (-1)^i \frac{\Gamma(\frac{1}{2}+n-i)}{i!(2n-i)!}(2x)^{n-2i} \tag{A.14}$$

The roots of this function are all real numbers in the interval $(-1, 1)$ and moreover it holds that

$$-1 < x_1^K < x_1^G < \cdots < x_n^K < x_n^G < x_{n+1}^K < 1.$$

We denote the Gaussian estimate of the integral as $I^G(f)$ and the Gauss-Kronrod estimate as $I^{GK}(f)$. A quadrature algorithm is called adaptive if an estimated error term is determined during run time of the algorithm and if it exceeds a predefined tolerance Δ, the integral is recursively calculated on $\left(a, \frac{a+b}{2}\right)$ and $\left(\frac{a+b}{2}, b\right)$, with a tolerance of $\frac{\Delta}{2}$. A simple version of this kind of algorithm is shown below:

Algorithm A.3 (adaptive Gauss-Kronrod-quadrature algorithm)

integrate(f, a, b, Δ)

Use $(A.11)$, $(A.12)$ to compute I^G and $(A.13)$, $(A.14)$ to compute I^{GK}

Calculate $\tilde{\varepsilon}^{GK} = |I^{GK} - I^G|$

if $\tilde{\varepsilon}^{GK} > \Delta$

$\int_a^b f(x)\mathrm{d}x = \text{integrate}(f, a, \frac{a+b}{2}, \frac{\Delta}{2}) + \text{integrate}(f, \frac{a+b}{2}, b, \frac{\Delta}{2})$

else

$\int_a^b f(x)\mathrm{d}x = I^{GK}$

end if

B. Matlab Code

Classical Model

This section contains the MATLAB files we use for Section 3.1. We start with the deterministic cost function.

```
function F = costfct_det(yz, mu, sigma, k1, k2, c_h, c_b)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%Function calculates the value of the costfucntion of a %%%%%%
%%%drifted Brownian motion intentory model with           %%%%%%
%%%deterministic supply.                                   %%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    c1 = @(y, z, k1, k2)  k1 + k2.*(z-y);
    g0_pos = @(x, mu,sigma, c_h) (x >=0).* ((c_h./(2.*mu)) .* x.^2 + (sigma.^2.*c_h.*x)./(2.*mu.^2));
    g0_neg = @(x, mu, sigma, c_h, c_b) (x<0).*((-c_b.*x.^2)./(2.*mu) - (sigma.^2.*c_b.*x)./(2.*mu.^2) +...
        (sigma.^4.*(c_b + c_h).*(exp((2.*mu.*x)./sigma.^2) - 1))./(4.*mu.^3));
    g0 = @(x, mu, sigma, c_h, c_b) g0_pos(x, mu, sigma, c_h) + g0_neg(x, mu, sigma, c_h, c_b);
    xi = @(x, mu) x./mu;
```

```
    F = (c1(yz(1), yz(2), k1, k2) + g0(yz(2), mu, sigma, c_h, c_b)
        - g0(yz(1), mu, sigma, c_h, c_b))./...
        (xi(yz(2), mu) - xi(yz(1), mu));
end
```

Cost function for a uniform distributed random supply.

```
function F = costfct_uniform(yz, mu, sigma, k1, k2, c_h, c_b, a)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%Function calculates the value of the costfucntion of a%%%%%%%
%%%Drifted Brownian motion intentory model with random   %%%%%%%
%%%supply, uniformly distributed on [p_a, z]            %%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if a < 1
    a_ratio = @(y, z, a) a.*z + (1-a).*y;
    c1 = @(y, z, k1, k2, a) k1 + k2.*((z.^2 - a_ratio(y, z, a).^2)
        ./(2.*(1-a).*(z-y)) - y);
    if a_ratio(yz(1), yz(2), a) >= 0
        aux1_g0 = @(y, z, mu, c_h, a) (c_h .* (z.^3 - a_ratio(y, z
            , a).^3))./...
            (6.*mu.*(1-a).*(z-y));
        aux2_g0 = @(y, z, mu, sigma, c_h, a) (sigma.^2.*c_h.*(z.^2
            - a_ratio(y, z, a).^2))./...
            (4.*mu.^2.*(1-a).*(z-y));
        g0_y_z = @(y, z, mu, sigma, c_h, c_b, a) aux1_g0(y, z, mu,
            c_h, a) + ...
            aux2_g0(y, z, mu, sigma, c_h, a);
    elseif yz(2) <= 0
       aux1_g0 = @(y, z, mu, c_b, a) (-c_b.*(z.^3 - a_ratio(y, z,
```

```
        a).^3))./...
          (6.*mu.*(1-a).*(z-y));
      aux2_g0 = @(y, z, mu, sigma, c_b, a) (-sigma.^2.*c_b.*(z.^2
          - a_ratio(y, z, a).^2))./...
          (4.*mu.^2.*(1-a).*(z-y));
      aux_aux1_g0 = @(mu, sigma, c_h, c_b) (sigma.^4 .*(c_b + c_h
        ))./(4.*mu.^3);
      aux_aux2_g0 = @(y, z, mu, sigma, a) sigma.^2/(2.*mu.*(1-a)
        .*(z-y));
      aux_aux3_g0 = @(z, mu, sigma) exp((2.*mu.*z)./sigma.^2);
      aux_aux4_g0 = @(y, z, mu, sigma) exp((2.*mu.*a_ratio(y, z,
        a))./sigma.^2);
      aux3_g0 = @(y, z, mu, sigma, c_h, c_b, a) aux_aux1_g0(mu,
        sigma, c_h, c_b).*...
          (aux_aux2_g0(y, z, mu, sigma, a).*(aux_aux3_g0(z, mu,
            sigma) - aux_aux4_g0(y, z, mu, sigma)) - 1);
      g0_y_z = @(y, z, mu, sigma, c_h, c_b, a) aux1_g0(y, z, mu,
        c_b, a) + ...
          aux2_g0(y, z, mu, sigma, c_b, a) + aux3_g0(y, z, mu,
            sigma, c_h, c_b, a);
  elseif yz(2) > 0 && a_ratio(yz(1), yz(2), a) < 0
      aux1_g0 = @(y, z, mu, c_h, c_b, a) (c_h.*z.^3 + c_b.*
        a_ratio(y, z, a).^3)./(6.*mu.*(1-a).*(z-y));
      aux2_g0 = @(y, z, mu, sigma, c_h, c_b, a) (sigma.^2.*(c_h.*
        z.^2 + c_b .*a_ratio(y, z, a).^2))./(4.*mu.^2.*(1-a).*(z
        -y));
      aux_aux1_g0 = @(y, z, mu, sigma, c_h, c_b, a) (sigma.^4 .*(
```

```
    c_b + c_h))./(4.*mu.^3.*(1-a).*(z-y));
  aux_aux2_g0 = @(mu, sigma) sigma.^2/(2.*mu);
  aux_aux3_g0 = @(y, z, mu, sigma) 1 - exp((2.*mu.*a_ratio(y,
      z, a))./sigma.^2);
  aux3_g0 = @(y, z, mu, sigma, c_h, c_b, a) aux_aux1_g0(y, z,
      mu, sigma, c_h, c_b, a).*...
      (aux_aux2_g0(mu, sigma) .* aux_aux3_g0(y, z, mu, sigma)
          + a_ratio(y, z, a));
  g0_y_z = @(y, z, mu, sigma, c_h, c_b, a) aux1_g0(y, z, mu,
    c_h, c_b, a) + ...
      aux2_g0(y, z, mu, sigma, c_h, c_b, a) - aux3_g0(y, z,
          mu, sigma,c_h, c_b, a);
end
if yz(1) >= 0
  g0_y = @(y, mu, sigma, c_h, c_b, a) (c_h .*y.^2)./(2.*mu) +
      ...
      (sigma.^2.*c_h.*y)./(2.*mu.^2);
elseif yz(1) < 0
  g0_y = @(y, mu, sigma, c_h, c_b, a) (-c_b .* y.^2)./(2.*mu)
      - ...
      (sigma.^2.*c_b.*y)./(2.*mu.^2) + sigma.^4.*(c_b + c_h)
          ./(4.*mu.^3) .*...
      (exp((2.*mu.*y)./sigma.^2) - 1);
end
xi_z_y = @(y, z, mu, a) (z.^2 - a_ratio(y, z, a).^2)./(2.*mu
  .*(1-a).*(z-y));
xi_y = @(y, mu) y./mu;
```

```
    F = (c1(yz(1), yz(2), k1, k2, a) + g0_y_z(yz(1), yz(2), mu, sigma, c_h, c_b, a) - ...
        g0_y(yz(1), mu, sigma, c_h, c_b, a))./(xi_z_y(yz(1), yz(2), mu, a) - xi_y(yz(1), mu));
elseif a == 1
    F = costfct_det(yz, mu, sigma, k1, k2, c_h, c_b);
end
```

Cost function for a binomial distributed random supply

```
function F = costfct_binomial(yz, mu, sigma, k1, k2, c_h, c_b, p, n)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%Function calculates the value of the costfucntion of a%%%%%%
%%%drifted Brownian motion intentory model with random   %%%%%%
%%%supply, binomial distributed on q_{k,y,z}            %%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    if p == 1
        F = costfct_det(yz, mu, sigma, k1, k2, c_h, c_b);
    else
        c1 = @(y, z, k1, k2)  k1 + k2.*(z-y);
        yz_r = @(y, z, k, n)  (n - k + 1).*y./(n+1) + k.*z/(n+1);
        g0_pos = @(x, mu,sigma, c_h) (x >=0).* ((c_h./(2.*mu)) .* x.^2 + (sigma.^2.*c_h.*x)./(2.*mu.^2));
        g0_neg = @(x, mu, sigma, c_h, c_b) (x<0).*((-c_b.*x.^2)./(2.*mu) - (sigma.^2.*c_b.*x)./(2.*mu.^2) +...
            (sigma.^4.*(c_b + c_h).*(exp((2.*mu.*x)./sigma.^2) - 1))./(4.*mu.^3));
        g0 = @(x, mu, sigma, c_h, c_b) g0_pos(x, mu, sigma, c_h) +
```

```
            g0_neg(x, mu, sigma, c_h, c_b);
        xi = @(x, mu) x./mu;
        prob = zeros(n+1, 1);
        for i=0:n
            prob(i+1, 1) = nchoosek(n, i) .* p.^i .*(1-p).^(n-i);
        end
        c1_bin = @(y, z, k1, k2, n) 0;
        g0_bin = @(y, z, mu, sigma, c_h, c_b, n) 0;
        xi_bin = @(y, z, mu, n) 0;
        for i=0:n
            c1_bin = @(y, z, k1, k2, n) c1_bin(y, z, k1, k2, n) +
                prob(i+1, 1) .*...
                 c1(yz(1), yz_r(y, z, i+1, n), k1, k2);
            g0_bin = @(y, z, mu, sigma, c_h, c_b, n) g0_bin(y, z, mu
                , sigma, c_h, c_b, n) +...
                prob(i+1, 1) .* g0(yz_r(y, z, i+1, n), mu, sigma, c_h
                    , c_b);
            xi_bin = @(y, z, mu, n) xi_bin(y, z, mu, n) + ...
                prob(i+1, 1) .* xi(yz_r(y, z, i+1, n), mu);
        end
        F = (c1_bin(yz(1), yz(2), k1, k2, n) + g0_bin(yz(1), yz(2),
             mu, sigma, c_h, c_b, n) ...
             - g0(yz(1), mu, sigma, c_h, c_b))./(xi_bin(yz(1), yz(2)
                 , mu, n) - xi(yz(1), mu));
    end
end
```

Cost function for a beta distributed supply

```
function F = costfct_beta(yz, mu, sigma, k1, k2, c_h, c_b, a, p, q)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%Function calculates the value of the costfucntion of a%%%%%%%
%%%Drifted Brownian motion intentory model with random   %%%%%%%
%%%supply, bet distributed on [p_a, z]                   %%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if a < 1
    a_ratio = @(y, z, a) a.*z + (1-a).*y;
    c_den_beta = @(y, z, a, p, q) 1./((gamma(p).*gamma(q).*...
        (z - a_ratio(y, z, a)).^(p+q-1))./gamma(p+q));
    den_beta = @(x, y, z, a, p, q) (x  - a_ratio(y, z, a)).^(p-1).*...
        (z - x).^(q-1);
    c1_int = @(y, x, k1, k2) k1 + k2*(x - y);
    c1 = @(y, z, k1, k2, a, p, q) k1 + c_den_beta(y, z, a, p, q).*...
        integral(@(x)den_beta(x, y, z, a, p, q).*k2.*(x - y), a_ratio(y, z, a), z);
    g0_pos = @(x, mu,sigma, c_h) (x >=0).* ((c_h./(2.*mu)) .* x.^2 + (sigma.^2.*c_h.*x)./(2.*mu.^2));
    g0_neg = @(x, mu, sigma, c_h, c_b) (x<0).*((-c_b.*x.^2)./(2.*mu) - (sigma.^2.*c_b.*x)./(2.*mu.^2) +...
        (sigma.^4.*(c_b + c_h).*(exp((2.*mu.*x)./sigma.^2) - 1))./(4.*mu.^3));
    g0_pn = @(x, mu, sigma, c_h, c_b) g0_pos(x, mu, sigma, c_h) + g0_neg(x, mu, sigma, c_h, c_b);
```

```
    xi = @(x, mu) x./mu;
    if a_ratio(yz(1), yz(2), a) >= 0
      g0_z = @(y, z, mu, sigma, c_h, c_b, a, p, q) c_den_beta(y, z, a, p, q).*...
        integral(@(x) den_beta(x, y, z, a, p, q).*g0_pos(x, mu, sigma, c_h), ...
        a_ratio(y, z, a), z);
    elseif yz(2) < 0
        g0_z = @(y, z, mu, sigma, c_h, c_b, a, p, q) c_den_beta(y, z, a, p, q).*...
        integral(@(x) den_beta(x, y, z, a, p, q).*g0_neg(x, mu, sigma, c_h, c_b), ...
        a_ratio(y, z, a), z);
    elseif yz(2) > 0 && a_ratio(yz(1), yz(2), a) < 0
        g0_z = @(y, z, mu, sigma, c_h, c_b, a, p, q) c_den_beta(y, z, a, p, q).*...
        (integral(@(x) den_beta(x, y, z, a, p, q).*g0_pos(x, mu, sigma, c_h), ...
        0, z) +integral(@(x) den_beta(x, y, z, a, p, q).*g0_neg(x, mu, sigma, c_h, c_b), ...
        a_ratio(y, z, a), 0));
    end
    if yz(1) >= 0
       g0_y = @(y, mu, sigma, c_h, c_b, a) (c_h .*y.^2)./(2.*mu) + ...
          (sigma.^2.*c_h.*y)./(2.*mu.^2);
    elseif yz(1) < 0
```

```
            g0_y = @(y, mu, sigma, c_h, c_b, a) (-c_b .* y.^2)./(2.*mu) - ...
                (sigma.^2.*c_b.*y)./(2.*mu.^2) + sigma.^4.*(c_h + c_h)./(4.*mu.^3) .*...
                (exp((2.*mu.*y)./sigma.^2) - 1);
        end
        xi_y_z = @(y, z, mu, a, p, q) c_den_beta(y, z, a, p, q).* ...
            (integral(@(x) den_beta(x, y, z, a, p, q).* xi(x, mu),...
            a_ratio(y, z, a),z)) - xi(y, mu);
        F = (c1(yz(1), yz(2), k1, k2, a, p, q) + g0_z (yz(1), yz(2),...
            mu, sigma, c_h, c_b, a, p, q) -  g0_y (yz(1), mu, sigma, c_h, c_b, a))./...
            (xi_y_z(yz(1), yz(2), mu, a, p, q));
elseif a == 1
        F = costfct_det(yz, mu, sigma, k1, k2, c_h, c_b);
end
```

All cost functions also were tested for correctness by using values close to 1 for a resp. p, which is nearly a deterministic supply. For the sake of avoiding repetition we only the routines for this section is shown. These can easily be changed for the following sections.

```
%test routine for the uniform cost function
%in a drifted BM model
clearvars
mu = 3;
sigma = 1.5;
k1 = 2;
k2 = 0.5;
```

```
c_h = 10;
c_b = 11;
a = 0.9999999999999;
yz = [-3 -1];
test1 = costfct_det(yz, mu, sigma, k1, k2, c_h, c_b) -
    costfct_uniform(yz, mu, sigma, k1, k2, c_h, c_b, a);
yz = [2 4];
test2 = costfct_det(yz, mu, sigma, k1, k2, c_h, c_b) -
    costfct_uniform(yz, mu, sigma, k1, k2, c_h, c_b, a);
a = 0.99999999999999;
yz = [-2000 0.001];
test3 = costfct_det(yz, mu, sigma, k1, k2, c_h, c_b) -
    costfct_uniform(yz, mu, sigma, k1, k2, c_h, c_b, a);
```

```
%test routine for the binomial cost function
%in a drifted BM model
clear all
p = 0.9999999;
yz = [-3 3];
mu = 2;
sigma = exp(-1);
k1 = 9;
k2 = 3;
c_h = 15;
c_b = 1;
n = 10;
test = costfct_binomial(yz, mu, sigma, k1, k2, c_h, c_b, p, n) -
    ...
```

```
    costfct_det(yz, mu, sigma, k1, k2, c_h, c_b)
```

The following functions calculate the expected value of the different supply distributions (they were also used to calculate the expected value in the later chapter and are therefore only stated once here:

```
function expec = exp_value_uniform_supply(yz, a)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%Function calculates the expected supply in a         %%%%%%%
%%%% intentory model with random supply uniform          %%%%%%%
%%%%distributed on [p_a, z]                              %%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    y = yz(1);
    z = yz(2);
    a_ratio = a*z + (1-a)*y;
    expec = (z + a_ratio)/2;
end
```

```
function expec = costfct_binomial_expectation(yz, p, n)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%Function calculates the expected supply in a         %%%%%%%
%%%%drifted Brownian motion intentory model with random   %%%%%%%
%%%%supply, binomial distributed on q_{k,y,z}            %%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    if p == 1
        expec = yz(2);
    else
        expec = 0;
        yz_r = @(y, z, k, n)  (n - k + 1).*y./(n+1) + k.*z/(n+1);
```

```
        prob = zeros(n+1, 1);
        for i=0:n
            prob(i+1, 1) = nchoosek(n, i) .* p.^i .*(1-p).^(n-i);
            expec = expec + prob(i+1)*yz_r(yz(1), yz(2), i+1, n);
        end
    end
end
```

```
function expec = exp_value_beta_supply(yz, a, p, q)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%Function calculates the expected supply in a      %%%%%%%
%%%% intentory model with random supply beta          %%%%%%%
%%%%distributed on [p_a, z]                           %%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    y = yz(1);
    z = yz(2);
    a_ratio = a*z + (1-a)*y;
    expec = (z - a_ratio)*p/(p+q) + a_ratio;
end
```

The following scripts were used to create the plots, which can be found in the corresponding chapter, again they are similar for all models, so to go not beyond the size of the thesis it is only stated here once.

```
%script to plot thrsholds, expected inventory level after supply
%and average cost in a drifted BM inventory model with uniform
%distributed supply
clearvars
clf
```

```
mu = 3;
sigma = 1;
k1 = 8;
k2 = 2;
c_h = 3;
c_b = 4;
numberOfa = 500;
x = zeros(2, numberOfa);
fval = zeros(1, numberOfa);
expec = zeros(1, numberOfa);
x0=[-3 3];
A = [1 -1];
b = 0;
a = linspace(0,1,numberOfa);
options = optimoptions('fmincon','Algorithm','sqp');
for j=1:numberOfa
    F = @(yz) costfct_uniform(yz, mu, sigma, k1, k2, c_h, c_b, a(j
        ));
    [x(:,j), fval(1, j)] = fmincon(F,x0,A,b,[],[],[],[],[],
        options);
    expec(1, j) = exp_value_uniform_supply(x(:, j), a(j));
    x0 = x(:,j);
end
figure(1)
subplot(2, 2, 4)
plot(a, fval, 'Linewidth', 1.5)
ylabel('average cost')
```

```
xlabel('a (interval of possible supply)')
subplot(2, 2, 3)
plot(a, expec, 'Linewidth', 1.5)
ylabel('expected inventory size after a order')
xlabel('a (interval of possible supply)')
subplot(2, 2, 1)
plot(a, x(1,:), 'Linewidth', 1.5)
ylabel('lower threshold')
xlabel('a (interval of possible supply)')
subplot(2, 2, 2)
plot(a, x(2,:), 'Linewidth', 1.5)
ylabel('upper threshold')
xlabel('a (interval of possible supply)')
```

```
%script to plot the average cost and
%the thresholds against four model
%parameters
clf
clearvars
mu = 3;
sigma = 1;
k1 = 8;
k2 = 2;
c_h = 3;
c_b = 4;
a1 = 0;
a2 = 1/2;
a3 = 3/4;
```

```
numberOfa = 500;
x = zeros(2, numberOfa);
y = zeros(24, numberOfa);
fval = zeros(12, numberOfa);
x0=[-0.01 0.01];
x0_a = [-3 3];
x0_b = [-3, 3];
A = [1 -1];
b = 0;
options = optimoptions('fmincon','Algorithm','sqp');
mu = linspace(0.05, 20, numberOfa);
for j=1:numberOfa
     if j < 10
        %for small mu, fmincon calculates
        %(y,z) in a way that z = y + k*10^(-10)
        %for which are Matlab calculates
        %average cost of -10^(-18), which
        %is obviously an error, which arise from
        %rounding error, therefore for these mu
        %b is set to -0.001, which has the effect
        %that z >= y + 0.001
        b = -0.001;
    end
    F = @(yz) costfct_uniform(yz, mu(j), sigma, k1, k2, c_h, c_b,
        a1);
    [x(:,j), fval(1, j)] = fmincon(F,x0,A,b,[],[],[],[],[],
        options);
```

```
        x0 = x(:,j);
        y(1, j) = x(1, j);
        y(2, j) = x(2, j);
        F = @(yz) costfct_uniform(yz, mu(j), sigma, k1, k2, c_h, c_b,
            a2);
        [x(:,j), fval(2, j)] = fmincon(F,x0_a,A,b,[],[],[],[],[],
            options);
        x0_a = x(:,j);
        y(3, j) = x(1, j);
        y(4, j) = x(2, j);
        F = @(yz) costfct_uniform(yz, mu(j), sigma, k1, k2, c_h, c_b,
            a3);
        [x(:,j), fval(3, j)] = fmincon(F,x0_b,A,b,[],[],[],[],[],
            options);
        x0_b = x(:,j);
        y(5, j) = x(1, j);
        y(6, j) = x(2, j);
end
mu = 3;
k1 = linspace(0.1, 20, numberOfa);
x0=[-3 3];
x0_a = [-3 3];
x0_b = [-3, 3];
for j=1:numberOfa
        F = @(yz) costfct_uniform(yz, mu, sigma, k1(j), k2, c_h, c_b,
            a1);
        [x(:,j), fval(4, j)] = fmincon(F,x0,A,b,[],[],[],[],[],
```

```
        options);
    x0 = x(:,j);
    y(7, j) = x(1, j);
    y(8, j) = x(2, j);
    F = @(yz) costfct_uniform(yz, mu, sigma, k1(j), k2, c_h, c_b, a2);
    [x(:,j), fval(5, j)] = fmincon(F,x0_a,A,b,[],[],[],[],[], options);
    x0_a = x(:,j);
    y(9, j) = x(1, j);
    y(10, j) = x(2, j);
    F = @(yz) costfct_uniform(yz, mu, sigma, k1(j), k2, c_h, c_b, a3);
    [x(:,j), fval(6, j)] = fmincon(F,x0_b,A,b,[],[],[],[],[], options);
    x0_b = x(:,j);
    y(11, j) = x(1, j);
    y(12, j) = x(2, j);
end
k1 = 8;
k2 = linspace(0.1, 20, numberOfa);
x0=[-3 3];
x0_a = [-3 3];
x0_b = [-3, 3];
for j=1:numberOfa
    F = @(yz) costfct_uniform(yz, mu, sigma, k1, k2(j), c_h, c_b, a1);
```

```
    [x(:,j), fval(7, j)] = fmincon(F,x0,A,b,[],[],[],[],[],
        options);
    x0 = x(:,j);
    y(13, j) = x(1, j);
    y(14, j) = x(2, j);
    F = @(yz) costfct_uniform(yz, mu, sigma, k1, k2(j), c_h, c_b,
        a2);
    [x(:,j), fval(8, j)] = fmincon(F,x0_a,A,b,[],[],[],[],[],
        options);
    x0_a = x(:,j);
    y(15, j) = x(1, j);
    y(16, j) = x(2, j);
    F = @(yz) costfct_uniform(yz, mu, sigma, k1, k2(j), c_h, c_b,
        a3);
    [x(:,j), fval(9, j)] = fmincon(F,x0_b,A,b,[],[],[],[],[],
        options);
    x0_b = x(:,j);
    y(17, j) = x(1, j);
    y(18, j) = x(2, j);
end
k2 = 2;
c_h = linspace(0.1, 20, numberOfa);
x0=[-3 3];
x0_a = [-3 3];
x0_b = [-3, 3];
for j=1:numberOfa
    F = @(yz) costfct_uniform(yz, mu, sigma, k1, k2, c_h(j), c_b,
```

```
        a1);
        [x(:,j), fval(10, j)] = fmincon(F,x0,A,b,[],[],[],[],[],options);
        x0 = x(:,j);
        b = 0;
        y(19, j) = x(1, j);
        y(20, j) = x(2, j);
        F = @(yz) costfct_uniform(yz, mu, sigma, k1, k2, c_h(j), c_b, a2);
        [x(:,j), fval(11, j)] = fmincon(F,x0_a,A,b,[],[],[],[],[],options);
        x0_a = x(:,j);
        %x0 = x0_a;
        y(21, j) = x(1, j);
        y(22, j) = x(2, j);
        F = @(yz) costfct_uniform(yz, mu, sigma, k1, k2, c_h(j), c_b, a3);
        [x(:,j), fval(12, j)] = fmincon(F,x0_b,A,b,[],[],[],[],[],options);
        x0_b = x(:,j);
        y(23, j) = x(1, j);
        y(24, j) = x(2, j);
end
figure(1)
subplot(2,2,1)
hold on
mu = linspace(0.05, 20, numberOfa);
```

```
plot(mu, fval(1,:), 'Linewidth', 1.5)
plot(mu, fval(2,:), 'Linewidth', 1.5)
plot(mu, fval(3,:), 'Linewidth', 1.5)
legend('a = 0', 'a= 1/2', 'a=3/4', 'Location', 'southeast')
ylabel('average cost')
xlabel('\mu')
subplot(2,2,2)
hold on
%all plots expect for the mu-plot are plotted
%against c_h, since c_h contains the same
%numbers, as k_1 and k_2
plot(c_h, fval(4,:), 'Linewidth', 1.5)
plot(c_h, fval(5,:), 'Linewidth', 1.5)
plot(c_h, fval(6,:), 'Linewidth', 1.5)
legend('a = 0', 'a= 1/2', 'a=3/4', 'Location', 'southeast')
ylabel('average cost')
xlabel('k_1')
subplot(2,2,3)
hold on
plot(c_h, fval(7,:), 'Linewidth', 1.5)
plot(c_h, fval(8,:), 'Linewidth', 1.5)
plot(c_h, fval(9,:), 'Linewidth', 1.5)
legend('a = 0', 'a= 1/2', 'a=3/4', 'Location', 'southeast')
ylabel('average cost')
xlabel('k_2')
subplot(2,2,4)
hold on
```

```
plot(c_h, fval(10,:), 'Linewidth', 1.5)
plot(c_h, fval(11,:), 'Linewidth', 1.5)
plot(c_h, fval(12,:), 'Linewidth', 1.5)
legend('a = 0', 'a= 1/2', 'a=3/4', 'Location', 'southeast')
ylabel('average cost')
xlabel('c_h')
figure(2)
hold on
subplot(2, 2, 1)
hold on
plot(mu, y(3,:), 'Linewidth', 1.5, 'Color', [0.4940 0.1840
    0.5560])
plot(mu, y(4,:), 'Linewidth', 1.5, 'Color', 'r')
ylabel('threshold')
xlabel('\mu')
subplot(2, 2, 2)
hold on
plot(c_h, y(9,:), 'Linewidth', 1.5, 'Color', [0.4940 0.1840
    0.5560])
plot(c_h, y(10,:), 'Linewidth', 1.5, 'Color', 'r')
ylabel('threshold')
xlabel('k_1')
subplot(2, 2, 3)
hold on
plot(c_h, y(15,:), 'Linewidth', 1.5, 'Color', [0.4940 0.1840
    0.5560])
plot(c_h, y(16,:), 'Linewidth', 1.5, 'Color', 'r')
```

```
ylabel('threshold')
xlabel('k_2')
subplot(2, 2, 4)
hold on
plot(c_h, y(21,:), 'Linewidth', 1.5, 'Color', [0.4940 0.1840
    0.5560])
plot(c_h, y(22,:), 'Linewidth', 1.5, 'Color', 'r')
ylabel('threshold')
xlabel('c_h')
%The following plots were not used in the thesis
figure(3)
hold on
plot(c_h, y(7,:),'.g')
plot(c_h, y(8,:),'.g')
plot(c_h, y(9,:),'.r')
plot(c_h, y(10,:),'.r')
plot(c_h, y(11,:),'.m')
plot(c_h, y(12,:),'.m')
figure(4)
hold on
plot(c_h, y(13,:),'.g')
plot(c_h, y(14,:),'.g')
plot(c_h, y(15,:),'.r')
plot(c_h, y(16,:),'.r')
plot(c_h, y(17,:),'.m')
plot(c_h, y(18,:),'.m')
figure(5)
```

```
hold on
plot(c_h, y(19,:),'.g')
plot(c_h, y(20,:),'.g')
plot(c_h, y(21,:),'.r')
plot(c_h, y(22,:),'.r')
plot(c_h, y(23,:),'.m')
plot(c_h, y(24,:),'.m')
figure(6)
hold on
plot(c_h, y(1,:),'.g')
plot(c_h, y(2,:),'.g')
plot(c_h, y(3,:), 'Linewidth', 1.5, 'Color', [0.4940 0.1840 0.5560])
plot(c_h, y(4,:), 'Linewidth', 1.5, 'Color', 'r')
plot(c_h, y(5,:),'.m')
plot(c_h, y(6,:),'.m')
ylabel('threshold')
xlabel('\mu')
```

Drifted Brownian motion with reflection at zero

This section contains the Code from Section 3.2, again we start with the cost function of a deterministic supply. Note that for the test scripts and the scripts to generate the plots see the corresponding scripts in the section before, which only need changed little to work for this model as well.

```
function F = costfct_det(yz, mu, sigma, k1, k2, k3, k4)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%Function calculates the value of the costfucntion of a%%%%%%%
%%%%drifted Brownian motion with reflection at 0          %%%%%%%
%%%%intentory model with deterministic supply              %%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    c1 = @(y, z, k1, k2) k1 + k2.*sqrt(z-y);
    g0 = @(x, mu, sigma, k3, k4) (k3.*x.^2)./(2.*mu) +...
        (k3.*sigma.^2.*x)./(2.*mu.^2) + (2.*k4.*sigma.^2.*(1-exp(-x)))./(sigma.^2.*(2.*mu+sigma.^2));
    xi = @(x, mu) x./mu;
    F = (c1(yz(1), yz(2), k1, k2) + g0(yz(2), mu, sigma, k3, k4) - g0(yz(1), mu, sigma, k3, k4))./...
        (xi(yz(2), mu) - xi(yz(1), mu));
end
```

Cost function for a uniform distributed supply

```
function F = costfct_uniform(yz, mu, sigma, k1, k2, k3, k4, a)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%Function calculates the value of the costfucntion of a%%%%%%%
%%%%drifted Brownian motion with reflection at 0          %%%%%%%
%%%%intentory model with random supply uniform           %%%%%%%
%%%% distributed on [q_a, z]                               %%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    if a < 1
        a_ratio = @(y, z, a) (1 - a) .* y + a.*z;
        c0 = @(y, z, k1, k2, a) k1 + (2.*k2.*(sqrt(z-y)-a.*sqrt(a.*(z-y)))).
/(3.*(1-a));
        aux1_g0_z_y = @(y, z, mu, k3, a) (k3 .* (z.^3 - a_ratio(y, z, a).^3))./(6.*mu.*(1-a).*(z-y));
```

```
        aux2_g0_z_y = @(y, z, mu, sigma, k3, a) (k3.*sigma.^2.*(z.^2 - a_ratio(y, z, a).^2)) ./...
            (4.*mu.^2.*(1-a).*(z-y));
        auxaux1 = @(mu, sigma, k4)  (2.*k4.*sigma.^2)./(sigma.^2.*(2.*mu + sigma.^2));
        auxaux2 = @(y, z, a) (exp(-z) - exp(-a_ratio(y, z, a)))./((1-a).*(z-y));
        aux3_g0_z_y = @(y, z, mu, sigma, k4, a) auxaux1(mu, sigma, k4) .* (1 +auxaux2(y, z, a));
        g0_z_y = @(y, z, mu, sigma, k3, k4, a) aux1_g0_z_y(y, z, mu, k3, a) + ...
            aux2_g0_z_y(y, z, mu, sigma, k3, a) + aux3_g0_z_y(y, z, mu, sigma, k4, a);

        g0_y = @(y, mu, sigma, k3, k4) (k3.*y.^2)/(2.*mu) + (k3.*sigma.^2.*y)./(2.*mu.^2) + ...
            (2.*k4.*sigma.^2.*(1-exp(-y)))./(sigma.^2.*(2.*mu + sigma.^2));

        xi_z_y = @(y, z, mu, a) (z.^2 - a_ratio(y, z, a).^2)./(2.*mu.*(1-a).*(z-y));
        xi_y = @(y, mu)  y./mu;
        F = (c0(yz(1), yz(2), k1, k2, a) + g0_z_y(yz(1), yz(2), mu, sigma, k3, k4, a) -...
            g0_y(yz(1), mu, sigma, k3, k4))./(xi_z_y(yz(1), yz(2), mu, a) - xi_y(yz(1), mu));
    elseif a == 1
```

```
        F = costfct_det(yz, mu, sigma, k1, k2, k3, k4);
    else
        msg = 'a must be smaller than 1';
        error(msg)
    end
end
```

Cost function for a polynomial distribution and a mixed polynomial distribution

```
function F = costfct_polynomial(yz, mu, sigma, k1, k2, k3, k4, a, k)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%Function calculates the value of the costfucntion of a%%%%%%
%%%drifted Brownian motion with reflection at 0          %%%%%%
%%%intentory model with random supply with a             %%%%%%
%%% polynomial distribution on [q_a, z]                  %%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    if a < 1
        %up-front auxiliary calculation
        a_ratio = @(y, z, a) a.*z + (1-a).*y;
        log_diff = @(y, z, a)  log(z) - log(a_ratio(y, z, a));
        if k == -1
            c = @(y, z, a, k) 1./log_diff(y, z, a) + (k -k);
        else
            c = @(y, z, a, k) (k+1)./(z.^(k+1) - a_ratio(y, z, a)^(k+1));
        end
        %costfunction c1
        integrator_c1 = @(y, k, v) sqrt(v-y) .* v.^k;
```

```
c1 = @(y, z, k1, k2, a, k) k1 + k2.*c(y, z, a, k) .* integral(@(v)integrator_c1(y, k, v), a_ratio(y, z, a), z);
%g0(y)
g0_y = @(y, mu, sigma, k3, k4) k3.*y.^2/(2.*mu) +...
    k3.*sigma.^2.*y./(2.*mu.^2) + (2.*k4.*sigma.^2.*(1-exp(-y)))./(sigma.^2.*(2.*mu + sigma.^2));
%g0(y,z) and its auxiliary functions
if k == -3
    aux1_g0 = @(y, z, mu, k3, a, k) (k-k) + c(y, z, a, k).*(k3.*log_diff(y, z, a))./(2.*mu);
else
    aux1_g0 = @(y, z, mu, k3, a, k) c(y, z, a, k) .*(k3.* (z.^(k+3) - a_ratio(y, z, a).^(k+3)))./(2.*mu.*(k+3));
end
if k == -2
    aux2_g0 = @(y, z, mu, sigma, k3, a, k) (k-k) + c(y, z, a, k) .*(k3.*sigma.^2.*(log_diff(y, z, a)))./(2.*mu.^2);
else
    aux2_g0 = @(y, z, mu, sigma, k3, a, k) c(y, z, a, k) .* k3.*sigma.^2.*(z.^(k+2) - (a.*z + (1-a).*y)^(k+2))./...
     (2.*mu.^2.*(k+2));
end
intergrator_g0 = @(k, v) exp(-v).*v.^k;
aux3_g0 = @(y, z, mu, sigma, k4, a, k) 2.*k4.*sigma
```

```
.^2.*(1 -...
            c(y, z, a, k) .* integral(@(v)intergrator_g0(k, v), a_ratio(y, z, a), z))./(sigma.^2.*(2.*mu + sigma.^2));
        g0_y_z = @(y, z, mu, sigma, k3, k4, a, k) aux1_g0(y, z, mu, k3, a, k) + ...
            aux2_g0(y, z, mu, sigma, k3, a, k) + aux3_g0(y, z, mu, sigma, k4, a, k);
        %xi(y,z) and xi(y)
        if k == -2
            xi_z_y = @(y, z, mu, a, k) (c(y, z, a, k) .* log_diff(y, z, a))./mu;
        else
            xi_z_y = @(y, z, mu, a, k) c(y, z, a, k) .* (z.^(k+2) - a_ratio(y, z, a).^(k+2))./((k+2).*mu); %((c(y, z, a) .*
        end
        xi_y = @(y, mu)  y./mu;
        %F(y, z)
        F = (c1(yz(1), yz(2), k1, k2, a, k) + g0_y_z(yz(1), yz(2), mu, sigma, k3, k4, a, k) - ...
            g0_y(yz(1), mu, sigma, k3, k4))./(xi_z_y(yz(1), yz(2), mu, a, k) - xi_y(yz(1), mu));
    elseif a == 1
        F = costfct_det(yz, mu, sigma, k1, k2, k3, k4);
    else
         msg = 'a must be smaller than 1';
```

```
            error(msg)
        end
end
```

```
function F = costfct_polynomial_mix(yz, mu, sigma, k1, k2, k3, k4, a, k_low, k_high, lambda)
    if a==1
        F = costfct_det(yz, mu, sigma, k1, k2, k3, k4);
    elseif k_low == k_high
        F = costfct_polynomial(yz, mu, sigma, k1, k2, k3, k4, a, k_low);
    else
        F = lambda*costfct_polynomial(yz, mu, sigma, k1, k2, k3, k4, a, k_low) + ...
            (1-lambda) * costfct_polynomial(yz, mu, sigma, k1, k2, k3, k4, a, k_high);
    end
end
```

The following function calculates the expected inventory after supply for a polynomial distributed supply

```
function expec = exp_value_poly_supply(yz, a, k)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%Function calculates the expected supply in a       %%%%%%
%%%% intentory model with random supply polynomial    %%%%%%
%%%distributed on [p_a, z]                            %%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    if a == 1
```

```
            expec = yz(2);
        else
            y = yz(1);
            z = yz(2);
            a_ratio = a.*z + (1-a).*y;
            if k == -1
                c = 1/(log(z) - log(a_ratio));
            else
                c = (k+1)/(z^(k+1) - a_ratio^(k+1));
            end
            if k == -2
                expec = c * (log(z) - log(a_ratio));
            else
                expec = (c*(z^(k+2) - a_ratio^(k+2)))/(k + 2);
            end
        end
end
```

Geometric Brownian motion

In this section includes the MATLAB Code from Section 3.3. As before we start with the cost function of a deterministic supply. As in the section before see for the scripts to test the functions and the scripts to produce the plots the first section of the Appendix, which contains examples, which can be used with small obvious changes to work for this section as well.

```
function F = costfct_det(yz, mu, sigma, k1, k2, k3, k4)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%Function calculates the value of the costfucntion of a%%%%%%
%%%drifted Brownian motion with reflection at 0          %%%%%%
%%%intentory model with deterministic supply             %%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    c1 = @(y, z, k1, k2)  k1 + k2.*sqrt(z-y);
    g0 = @(x, mu, sigma, k3, k4) (k3.*x.^2)./(2.*mu) +...
        (k3.*sigma.^2.*x)./(2.*mu.^2) + (2.*k4.*sigma.^2.*(1-exp(-x)))./(sigma.^2.*(2.*mu+sigma.^2));
    xi = @(x, mu) x./mu;
    F = (c1(yz(1), yz(2), k1, k2) + g0(yz(2), mu, sigma, k3, k4) - g0(yz(1), mu, sigma, k3, k4))./...
        (xi(yz(2), mu) - xi(yz(1), mu));
end
```

Cost function for a uniform distributed supply

```
function F = costfct_uniform(yz, mu, sigma, k1, k2, k3, k4, a)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%Function calculates the value of the costfucntion of a%%%%%%
%%%drifted Brownian motion with reflection at 0          %%%%%%
%%%intentory model with random supply uniform            %%%%%%
%%% distributed on [q_a, z]                              %%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    if a < 1
        a_ratio = @(y, z, a) (1 - a) .* y + a.*z;
        c0 = @(y, z, k1, k2, a) k1 + (2.*k2.*(sqrt(z-y)-a.*sqrt(a.*(z-y))))./(3.*(1-a));
        aux1_g0_z_y = @(y, z, mu, k3, a) (k3 .* (z.^3 - a_ratio(y, z, a).^3))./(6.*mu.*(1-a).*(z-y));
```

```
        aux2_g0_z_y = @(y, z, mu, sigma, k3, a) (k3.*sigma.^2.*(z.^2 - a_ratio(y, z, a).^2)) ./...
          (4.*mu.^2.*(1-a).*(z-y));
        auxaux1 = @(mu, sigma, k4)  (2.*k4.*sigma.^2)./(sigma.^2.*(2.*mu + sigma.^2));
        auxaux2 = @(y, z, a) (exp(-z) - exp(-a_ratio(y, z, a)))./((1-a).*(z-y));
        aux3_g0_z_y = @(y, z, mu, sigma, k4, a) auxaux1(mu, sigma, k4) .* (1 +auxaux2(y, z, a));
        g0_z_y = @(y, z, mu, sigma, k3, k4, a) aux1_g0_z_y(y, z, mu, k3, a) + ...
          aux2_g0_z_y(y, z, mu, sigma, k3, a) + aux3_g0_z_y(y, z, mu, sigma, k4, a);

        g0_y = @(y, mu, sigma, k3, k4) (k3.*y.^2)/(2.*mu) + (k3.*sigma.^2.*y)./(2.*mu.^2) + ...
          (2.*k4.*sigma.^2.*(1-exp(-y)))./(sigma.^2.*(2.*mu + sigma.^2));

        xi_z_y = @(y, z, mu, a) (z.^2 - a_ratio(y, z, a).^2)./(2.*mu.*(1-a).*(z-y));
        xi_y = @(y, mu)  y./mu;
        F = (c0(yz(1), yz(2), k1, k2, a) + g0_z_y(yz(1), yz(2), mu, sigma, k3, k4, a) -...
          g0_y(yz(1), mu, sigma, k3, k4))./(xi_z_y(yz(1), yz(2), mu, a) - xi_y(yz(1), mu));
    elseif a == 1
```

```
            F = costfct_det(yz, mu, sigma, k1, k2, k3, k4);
        else
            msg = 'a must be smaller than 1';
            error(msg)
        end
end
```

Cost function for a binomial distributed supply

```
function F = costfct_binomial(yz, mu, sigma, k1, k2, k3, k4, beta, p, n)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%Function calculates the value of the costfucntion of a%%%%%%
%%%geometric Brownian motion intentory model with random %%%%%%
%%%supply, binomial distributed on q_{k,y,z}            %%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    if beta > 0
        error('beta needs to be smaller than 0')
    end
    if p == 1
        F = costfct_det(yz, mu, sigma, k1, k2, k3, k4, beta);
    else
        rho = @(mu, sigma, beta) sigma^2.*beta.*(beta-1)/2 - mu.*beta;
        yz_r = @(y, z, k, n)  (n - k + 1).*y./(n+1) + k.*z/(n+1);
        c1 = @(y, z, k1, k2) k1 + k2.*sqrt(z -y);
        g_0 = @(x, mu, sigma, k3, k4, beta) k3./mu .*x - (k4./rho(mu, sigma, beta)).*x.^beta;
```

```
        xi = @(x, mu, sigma) (2./(2.*mu + sigma.^2)).*log(x);
        prob = zeros(n+1, 1);
        for i=0:n
          prob(i+1, 1) = nchoosek(n, i) .* p.^i .*(1-p).^(n-i);
        end
        c1_bin = @(y, z, k1, k2, n) 0;
        g0_bin = @(y, z, mu, sigma, k3, k4, beta, n) 0;
        xi_bin = @(y, z, mu, sigma, n) 0;
       for i=0:n
          c1_bin = @(y, z, k1, k2, n) c1_bin(y, z, k1, k2, n) + prob(i+1, 1) .*...
              c1(yz(1), yz_r(y, z, i+1, n), k1, k2);
          g0_bin = @(y, z, mu, sigma, k3, k4, beta, n) g0_bin(y, z, mu, sigma, k3, k4, beta, n) +...
                 prob(i+1, 1) .* g_0(yz_r(y, z, i+1, n), mu, sigma, k3, k4, beta);
          xi_bin = @(y, z, mu, sigma, n) xi_bin(y, z, mu, sigma, n) + ...
              prob(i+1, 1) .* xi(yz_r(y, z, i+1, n), mu, sigma);
       end
       F = (c1_bin(yz(1), yz(2), k1, k2, n) + g0_bin(yz(1), yz(2), mu, sigma, k3, k4, beta, n) ...
           - g_0(yz(1), mu, sigma, k3, k4, beta))./(xi_bin(yz(1), yz(2), mu, sigma, n) - xi(yz(1), mu, sigma));
    end
end
```

Cost function for a truncated normal distributed supply and a function to compute the

expected value after supply in the case of a truncated normal supply

```
function F = costfct_normal(yz, mu, sigma, k1, k2, k3, k4, beta, a, mu_norm, sigma_norm)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%Function calculates the value of the costfucntion of a%%%%%%%
%%%Geometric Brownian motion intentory model with random %%%%%%%
%%%supply, truncated normal distributed on [p_a, z]      %%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    if beta > 0
       error('beta needs to be smaller than 0')
    end
    if a == 1
        F = costfct_det(yz,  mu, sigma, k1, k2, k3, k4, beta);
    else
        a_ratio = @(y, z, a) a.*z + (1-a).*y;
        c_den_normal = @(y, z, a, mu_norm, sigma_norm) 1./(sigma_norm .*...
            (normcdf((z - mu_norm)/sigma_norm) - ...
            normcdf((a_ratio(y, z, a) - mu_norm)./sigma_norm))) ;
        den_normal = @(x, y, z, a, mu_norm, sigma_norm) c_den_normal(...
            y, z, a, mu_norm, sigma_norm).*normpdf((x - mu_norm)./sigma_norm);
        rho = @(mu, sigma, beta) sigma^2.*beta.*(beta-1)/2 - mu.*beta;
        c1 = @(y, z, k1, k2, a, mu_norm, sigma_norm) k1 + ...
```

```
                    integral(@(x)den_normal(x, y, z, a, mu_norm, sigma_norm).*k2.*sqrt(x - y), a_ratio(y, z, a), z);
        g0 = @(x, mu, sigma, k3, k4) k3.*x./mu -...
                k4.*x.^(beta)./rho(mu, sigma, beta);
        g0_z = @(y, z, mu, sigma, k3, k4, mu_norm, sigma_norm) ...
                integral(@(x) den_normal(x, y, z, a, mu_norm, sigma_norm).*...
                g0(x, mu, sigma, k3, k4), a_ratio(y, z, a), z);
        xi = @(x, mu, sigma) 2*log(x)./(2.*mu + sigma.^2);
        xi_z = @(y, z, mu, sigma, a, mu_norm, sigma_norm) integral(@(x) ...
                den_normal(x, y, z, a, mu_norm, sigma_norm).*xi(x, mu, sigma),...
                a_ratio(y, z, a), z);
        F = (c1(yz(1), yz(2), k1, k2, a, mu_norm, sigma_norm) + ...
                g0_z(yz(1), yz(2),mu, sigma, k3, k4, mu_norm, sigma_norm) - ...
                g0(yz(1), mu, sigma, k3, k4))./...
                (xi_z(yz(1), yz(2), mu, sigma, a, mu_norm, sigma_norm) - xi(yz(1), mu, sigma));
    end
end
```

```
function exp = exp_val_trun_norm(y, z, mu_trun, sigma_trun, a)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%Function calculates the expected supply in a          %%%%%%
%%%% intentory model with random supply truncated normal  %%%%%%
```

```
%%%%distributed on [p_a, z]                                   %%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    if a == 1
        exp = z;
    else
        a_ratio = (1 - a)*y + a*z;
        num = normpdf((a_ratio - mu_trun)/sigma_trun) - normpdf((z - mu_trun)/sigma_trun);
        den = normcdf((z - mu_trun)/sigma_trun) - normcdf((a_ratio - mu_trun)/sigma_trun);
        exp = mu_trun + sigma_trun*num/den;
    end
end
```