Submitted to *Canvas*

# EXPLORING THE FEASIBILITY OF ARBITRAGE IN THE SCARF ECONOMY

THOMAS VAN DER ZIJDEN

JUNE 2022

SUPERVISED BY AAD RUITER

The literature on financial trading agents is primarily focused on trading performance, and not necessarily on arbitrage. The experiments of Anderson et al. (2004) have shown that human traders engage in arbitrage. This paper aims to test if arbitrage is a viable strategy for trading agents in the Scarf economy. We will propose modifications to the eGD algorithm to test the profitability of an arbitrage strategy in the stable version of the Scarf economy. Results have shown that we were able to create a trading agent which could consistently make arbitrage profits. The trading agents were not able to gain extra utility from these arbitrage profits as they did not amount to enough extra capital.

KEYWORDS: Arbitrage, Agent Based Modelling, Scarf, Continuous Double Auction, Trading agents.

This document is written by Thomas van der Zijden who declares to take full responsibility for the contents of this document. I declare that the text and the work presented in this document are original and that no sources other than those mentioned in the text and its references have been used in creating it. UvA Economics and Business is responsible solely for the supervision of completion of the work and submission, not for the contents.

## 1. INTRODUCTION

The experiments of Anderson et al. (2004) have shown that human traders engage in arbitrage. How can we model this type of behavior? The literature on financial trading agents is primarily focused on trading performance, and not necessarily on arbitrage. Notable algorithms include ZIC, ZIP, GD, and AA. The GD algorithm is of particular interest, because it introduces beliefs as to the likelihood that orders will be accepted. These beliefs can be used to determine the feasibility of arbitrages. This paper will document the creation of an algorithm that looks for feasible arbitrage trades using the GD belief function. The main focus is finding the important factors that make arbitrage profitable.

## 2. THEORETICAL FRAMEWORK

To study the behaviour that governs financial markets Smith (1962) created a virtual market where buyers and sellers interacted with each other through double auction until feasible trades were exhausted. Smith (1962) proved with this experiment that a central auctioneer was not necessary for market outcomes to reach efficiency. Although this system was devised for human agents, the simple design of the experiment proved to be very useful for testing algorithmic agents.

Over the years, many algorithmic trading agents have been developed and tested for the Smith environment using continuous double auction. At first, it was believed that trading agents needed to be profit-seeking to reach market efficiency. Gode and Sunder (1993) have shown that even zero-intelligence traders that were not allowed to make losses can reach market efficiency in this system. This suggests that intelligence of the traders is not necessary for a competitive market to reach efficiency.

However, Cliff and Bruten (1997) showed that the ZIC traders of Gode and Sunder only reach market efficiency when demand and supply are symmetric but not necessarily when the market is asymmetric. To achieve convergence with asymmetric demand and supply, Cliff introduced a slightly more intelligent trading algorithm called *Zero Intelligence Plus* (ZIP). ZIP uses an internal profit margin which it increases or decreases depending on the most recent quote price and if it resulted in a trade or not. Cliff demonstrated that ZIP reached market efficiency in both symmetric and asymmetric configurations of the Smith market.

To maximize profits in the experimental markets following Smith's framework , other more intelligent algorithms have been developed. The most notable being GD named after its inventors Gjerstad and Dickhaut (1998) and the *Adaptive-Aggressive* (AA) algorithm made by Vytelingum (2006). GD maximises profits conditional on own beliefs by using historical data to estimate the probability of a trade succeeding and uses that information to select the best quote price. This algorithm has been extended by Tesauro and Bredin (2002), who used dynamic programming to optimize a long-term discounted cumulative profitability rather than immediate profit. They called their algorithm *eXtended GD* or GDX. The AA algorithm uses short and long-term learning to update its internal profit margin. In the short term, it updates this margin using rules similar to ZIP. Over the long-term, AA tries to estimate the market equilibrium by calculating a moving average of the historical transaction prices. It uses this estimate to calculate if it is trading intra or extra-marginal. If AA estimates it is extra-marginal it will trade more aggressively by decreasing its profit margin.

Until 2020, the research done with the Smith model was mainly focused on studying the pricing strategies of the trading agents but did not address the importance of timing. Hanifan et al. (2021) tested the importance of timing by re-evaluating the ranking using reaction speed and modified the ZIP algorithm to incorporate trading urgency. The algorithms were ranked in speed based on how fast they could compute their next move and faster algorithms could act faster and more often. Hanifan et al. (2021) found that intelligent algorithms (GDZ, AA), which were previously thought to be superior, now were outperformed by the simpler algorithms (ZIP, SHVR) because they had more opportunities to act.

Arbitrage played little to no role in the research done using the Smith environment since agents have no price expectations and each trading agent has a clear fixed role of buyer or seller.

This paper will study arbitrage in the experimental setup of Anderson et al. (2004). Here traders clearly engaged in arbitrage as they were shown to buy goods from which they derived no utility and they sold goods from which they did. The experiment uses the economy of Scarf (1960) which has three commodities and three types of traders. Every trader has its own unique Leontief utility function where they derive utility from two out of the three goods. This setup is appropriate because the Leontief utility function gives an obvious indication when a trader engages in arbitrage. This is exactly what the experiments of Anderson et al. (2004) have shown. Traders in the setup of Anderson have a short time to form price expectations. If their price expectation is higher than what the good is currently trading for it might lead them to buy a good they do not need with the intent of selling it back later at a profit. On the other hand if their price expectation is much lower than that it is currently trading for they might sell a good they need with the intent of buying it back later for cheaper.

Anderson et al. (2004) demonstrated through experiments that human traders reach the Walrasian equilibrium in the stable version of the famous Scarf (1960) economy. Ruiter (2018) took the data from the experiment of Anderson and tried to replicate it using various trading agents. By looking at the predictive quality of these algorithms and the way they calculate price expectation Ruiter tried to deduce how humans propose prices. Ruiter proposed modifications to the most popular trading algorithms from the Smith environment in order for them to operate in the setup of Anderson. The algorithm from Ruiter that is of most interest to this paper is a modification of the GD algorithm called *expected GD* (eGD).

This paper will propose some modifications to the eGD algorithm to test if arbitrage trades can be a viable strategy for a trading agent in the Scarf economy. Our hypothesis is that an arbitrage strategy can increase the average utility of a trader if it generates enough profit. If it can generate enough capital to afford an extra good it can increase its utility compared to a regular eGD trader. We also expect that the feasibility of arbitrage will critically depend on time: (i) if an arbitrage takes longer to complete, then the arbitrageur may have to compete with other traders, which will adversely affect profits from arbitrage; (ii) at the end of the game, traders do not want to own commodities that do not yield utility.

## 3. METHODOLOGY

### 3.1. *The Scarf economy*

The experiments of Anderson et al. (2004) used the economy of Scarf (1960) which consisted of three types of traders $i = 1,2,3$ and three commodities $j = 1,2,3$ where the first commodity serves the role of money. This creates two markets: one where good 1 can be exchanged for good 2 and one where good 1 can be exchanged for good 3. A bundle is denoted by $\mathbf{x} \in \mathbb{R}^3_+$ and $x_{ji}$ refers to the amount trader $i$ has of commodity $j$. Each trader has the following Leontief utility function describing its preferences:[1]

$$U^1(\mathbf{x}_1) = \min\left\{\tfrac{x_{11}}{400}, \tfrac{x_{31}}{20}\right\}$$
$$U^2(\mathbf{x}_2) = \min\left\{\tfrac{x_{12}}{400}, \tfrac{x_{22}}{10}\right\}$$
$$U^3(\mathbf{x}_3) = \min\left\{\tfrac{x_{23}}{10}, \tfrac{x_{33}}{20}\right\}$$

We will use the same initial endowments of the basic stable version[2] of the scarf economy as Anderson et al. (2004) have used in their human experiments so we can later compare performance. This allocation was:

$$\mathbf{w} = \begin{pmatrix} 0 & 0 & 400 \\ 10 & 0 & 0 \\ 0 & 20 & 0 \end{pmatrix}$$

Every trader type has a good in which its not interested. Buying a good in which the trader is not interested is a clear indication of arbitrage. This is because this move is only justified for a utility maximizing trader if the trader expects to sell it back for more. The same holds for a trader that sells a good from which it derived utility. The experiments from Anderson et al. (2004) clearly show that humans engage in this type of behaviour. Arbitrage behaviour consisted about $7.5\%$ of the total actions taken in session 414 of the experiment.

### 3.2. *Simulated CDA*

The algorithms will be trading in this economy using the rules of Continuous Double Auction (CDA), which is done through a platform developed specifically for this paper. In this simulated market, traders can submit orders to either buy or sell a good. An order to sell a good is called an ask and an order to buy a good is called a bid. Traders can only submit feasible orders, meaning they must have the resources necessary to complete the order once it is accepted by another trader. Orders are saved in an order book with depth one. This means that only the best ask and bid, also known as the floor order, is saved in the order book and if a better order is presented the previous one is lost. These orders can result in a transaction if they cross the floor

[1]The preferences, endowments and the rule that good is money comes from Anderson et al. (2004).

[2]Anderson showed that all allocations have the same theoretical equilibrium prices but the process of tatonnement does not reach that in the other allocations. Arbitrage needs good price expectations so the stable environment is the best for possible arbitrage. Although as mentioned by Anderson if the clockwise/anti-clockwise allocation has a predictable pattern it might be better for arbitrage.

price or it can replace the current floor order. When a trader posts an order that does no result in a trade or improves the floor order then it is ignored by the exchange. The acceptance of a floor order can be partial, e.g. if three units of commodity 3 are being sold, then a buyer can choose to buy just one unit leaving the remaining two units in the order book.

### 3.3. *Algorithms*

Most research that was done on algorithms in Smith's environment relied on a exogenous limit price of a single commodity from which expected profits could be calculated. The Scarf economy has two goods and traders want to gain a maximal amount of utility instead of a maximum amount of profit. Therefore there is no exogenous limit price but rather an endogenous one. This poses a problem when trying to convert the algorithms to the Scarf economy as their rules for updating parameters and posting orders rely on this exogenous limit price. Ruiter (2018) proposed a few modifications to these algorithms so they can operate in the Scarf economy. ZIP by Cliff and Bruten (1997) calculates the shout price of its next order by the formula $p^s_j = p^r_j(1 - \varphi_j)$. Where $p^s_j$ is the shout price , $p^r_j$ the reservation price and $\varphi_j$ an internal profit margin ($\varphi_j < 0$ for buyers and $\varphi_j > 0$ for sellers). The ZIP algorithm specifies the rules for updating this internal profit margin $\varphi_j$. Ruiter proposed that by using the ZIP algorithm to update shout prices instead of profit margins it is possible to apply it in the context of the Scarf economy. In our case ZIP does not use a markup ($\varphi_j = 0$) as it updates shouts prices rather than a profit margin. In our simulations ZIP chooses a random feasible order and has no sense of its own utility function. If a traders is not an arbitrageur then a feasible order is defined as an order where the trader has the funds to complete it and that it does not decrease its utility.

Ruiter also noted that we can use the belief function of Gjerstad and Dickhaut (1998), which estimates a probability that a bid or an ask is accepted at a given price, to calculate so called "no arbitrage" prices. The "no arbitrage" price is the price where the acceptance of a bid and ask at that price is equally likely. The belief function of Gjerstad and Dickhaut (1998) was calculated in the following way. The algorithm saves a history of accepted and rejected asks $a$ and bids $b$ in a set $\mathbb{H}$ for any given price $p_{bid}, p_{ask}$ the function below exists if the bid or ask

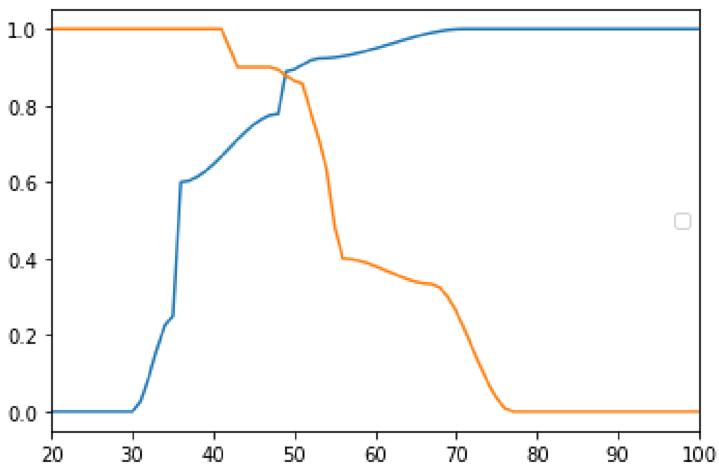



FIGURE 1.—GD belief curves. The jagged look of the curves is due to the fact that it has to interpolate between unobserved values. It creates an unique spline for the gaps between each non-consecutive integer values

was previously observed at that price. I.e $p_{bid} \in \mathbb{H}_b$ for bids and $p_{ask} \in \mathbb{H}_a$ for asks.

$$
\begin{aligned}
P\,[\text{ bid accepted at } p_{bid}] &= \frac{\sum\limits_{p \le p_{bid}} q_{bid}^{acc}(p) + \sum\limits_{p \le p_{bid}} q_{ask}(p)}{\sum\limits_{p \le p_{bid}} q_{bid}^{acc}(p) + \sum\limits_{p \le p_{bid}} q_{ask}(p) + \sum\limits_{p \ge p_{bid}} q_{bid}^{rej}(p)} \\
P\,[\text{ ask accepted at } p_{ask}] &= \frac{\sum\limits_{p \ge p_{ask}} q_{ask}^{acc}(p) + \sum\limits_{p \ge p_{ask}} q_{bid}(p)}{\sum\limits_{p \ge p_{ask}} q_{ask}^{acc}(p) + \sum\limits_{p \ge p_{ask}} q_{bid}(p) + \sum\limits_{p \le p_{ask}} q_{ask}^{rej}}
\end{aligned} \tag{1}
$$

For values that have not been observed before GD uses spline interpolation to fill out the missing values. We assume that that there is a max price $M$ such that $P\,[\text{ bid accepted at } M] = 1$ and $P\,[\text{ ask accepted at } M] = 0$. In our exchange $M = 200$. Due to the fact that we have to improve the order book the probability of posting a bid at or below floor bid or posting an ask at or above floor ask is $0$. This belief function can be used to calculate two curves and where they intersect is the "no-arbitrage" price. Figure 1 shows an example of what the curves look like. The intersection at $p = 49$ is the "no-arbitrage" price. This can be interpreted as the equilibrium price of the good and are used as the expectation price by the algorithm *expected Gjerstad Dickhaut* (eGD).

Different types of trading agents share the same two main functions:

- `get_order` Gives an order and is called when the trader is selected to submit an order.

- `respond` Updates the internal parameters of the trader and is called every time the order book changes either through a trade or improvement of the floor-price.

To simplify the behaviour of the algorithms we have chosen to restrict the quantity of each order to $1$. This alleviates the added complexity of not only choosing the optimal price for the order but also the optimal quantity.

### 3.4. *Arbitrage*

Normally arbitrage is defined as the practice of taking advantage of a price difference in two markets of the same good. The arbitrageur profits from buying a good on the cheaper market and selling it on the more expensive market. The arbitrage traders engage in the experiments of Anderson et al. (2004) is time arbitrage. The experiments of Anderson et al. (2004) showed that human traders of type $I$ and $II$ were posting orders to buy the commodity from which they do not derive utility with the intent of selling it back later for a profit. It was also seen that traders were selling goods they needed with the intent of buying them back later to restore their level of utility. The motivation behind this behaviour would be that if the trader believes the market is trading at a non equilibrium price, he can profit from the difference. Arbitrage requires two successful orders. First we have to sell a good we need or buy a good we do not (The strategic order). Second we have to buy it back for less or sell it back for more (The arbitrage order).

This paper will suggest its own variant of the eGD algorithm to spot potential arbitrage opportunities called *Gjerstad Dickhaut Zijden* (GDZ).

GDZ behaves the same way as eGD except for the fact that it has a $20\%$ probability to look at for an arbitrage opportunity every time step. If it is not able to find a potentially profitable arbitrage opportunity in that time step, it reverts back to the eGD behavior. GDZ looks for these arbitrage opportunities using the following logic. Let $\bar{p}_j^b$ and $\bar{p}_j^a$ be the floor bid and ask price for good $j$. For goods $j = 2, 3$ if prices $\bar{p}_j^b$ and $\bar{p}_j^a$ exist we can accept the bid or ask with probability $1$ and then calculate the probability of selling or buying it back later at a profit. If we intend to sell or buy the good back later for a profit we can interpret the floor price as the reservation price $p_j^r$. We use this reservation price together with the GD belief function to calculate expected profit and see what the optimal next order would be.[3] We restrict the strategic

[3]The reason we maximize expected profit and not utility is that from one single arbitrage trade the trader will not gain any utility. Because money can be exchanged for both goods if the trader can generate enough profits using arbitrage it will on average have more funds to spend and in the long term gain slightly more utility.

trade to have a probability of 1 to be accepted meaning that if we want we want to buy good 2 to sell it back later we need that $\bar{p}_2^a$ exists. If we do not restrict the strategic trade to being certain than we have to account for probability both orders being accepted which adds an extra layer of complexity. While this probably would be more profitable it is beyond the scope of this paper.

The optimal arbitrage order is calculated as follows: Say we are a trader looking at buying good $j$ to sell it back later. We check if floor ask exists and set it as the reservation price $p_j^r = \bar{p}_j^a$. We use the GD belief function to create a vector of probabilities of the arbitrage order being accepted given the price. This vector is defined as $\mathbf{v} : 200 \times 1$ where element $v_i = P[\text{ask accepted at } p = i]$. Next we create a profit vector $\mathbf{a} : 200 \times 1$ where element $a_i = \max(0, i - p_j^r)$ Let $\odot$ denote element wise multiplication for vectors. We can calculate the optimal price $p_j^{opt}$ by getting the price for which the expected profit is highest.

$$p_j^{opt} = \arg\max_p(\mathbf{v} \odot \mathbf{a})$$

After calculating $p_j^{opt}$ we post the strategic order to buy good for $\bar{p}_j^a$ and save the arbitrage order to ask $p_j^{opt}$ for good $j$ to post in our next turn. For selling a good we want to buy back later the process is analogous. The only difference is $p_j^r = \bar{p}_j^b$ and the elements of the profit vector are now $a_i = \max(0, p_j^r - i)$ . If there were multiple arbitrage opportunities then the algorithm chose the one with the highest expected profit. E.g trader of type $III$ can sell either good 2 or 3 as an arbitrage opportunity and will choose which opportunity to engage in based on what gives him the highest amount of expected profit. What if their price expectation when calculating the trade was not correct and the trade is rejected? The algorithm responds to this rejection by lowering its profit margin by undercutting the competitor by 1. E.g if his bid of 50 is rejected by a bid of 48 its next order will be a bid for 47 even if that means taking a loss on the product.

Preliminary testing has shown a few issues with arbitrage. These issues were:

- The GD belief function overestimates the probability that goods can be sold at prices that are far from the prices goods were previously traded at.
- GDZ took too much risk when choosing the price with the highest expected utility.
- Arbitrage orders were rejected before the trader had the opportunity to post them.

These issues were addressed in the following ways. The first issue is that the GD belief function overestimates the probabilities in the extremes. The GD belief function estimates probability of a bid being accepted conditional on observing an ask. Ruiter proposed that a better belief

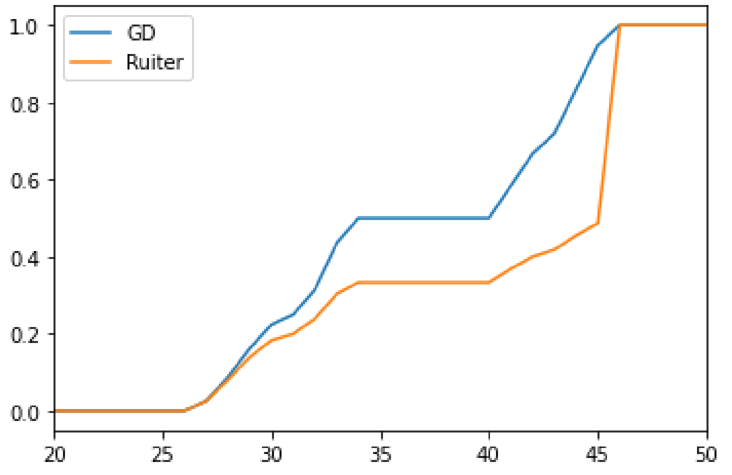



FIGURE 2.—Example of an estimated probability of a bid being accepted at different prices.

function is the unconditional probability a bid will be accepted before it is rejected. The formula for bid being accepted at $\bar{p}_j^b$:

$$B\left[\bar{p}_j^b\right] = \frac{P_A\left[p^a \leq \bar{p}_j^b\right]}{P_A\left[p^a \leq \bar{p}_j^b\right] + P_B\left[p^b > \bar{p}_j^b\right]} \tag{2}$$

As can be seen in figure 2 the Ruiter belief function gives a more conservative estimation probability the bid will be accepted for prices that stray far from the observed prices.

The second issue is that GDZ tend to choose optimal prices $p^{opt}$ that have a very low probability of succeeding since the expected profit was still highest. These unrealistic profits led to very high wait times between the strategic trade and the eventual successful arbitrage trade. To circumvent this we introduce the restriction that arbitrage traders can only select optimal prices which satisfy $P[\text{order accepted at } p_j^{opt}] \geq \theta_{min}$ where $0 < \theta_{min} < 1$. This restriction is imposed by setting all probabilities that do not satisfy this inequality to 0. So we replace $\mathbf{v}$ with $\hat{\mathbf{v}}$ where

$$\hat{v}_i = \begin{cases} v_i \; v_i \geq \theta_{\text{min}} \\ 0 \quad v_i < \theta_{\text{min}} \end{cases}$$

The last issue is that arbitrage orders were rejected before the trader could post them in the next time step. E.g a trader has bought good 2 for 45 with the intent of selling it for 50 it is rejected before it can even be posted if another trader posts an ask for 48 in the meantime or even worse for 44. This forces the trader to make a loss before it even had the time to post its next move. To solve this we allow the trader to immediately post the arbitrage order after its strategic order without waiting for the next time step. This decision is justified by the fact that in real life arbitrage traders quicker than others to capitalize on the price difference between markets.

### 3.5. *Simulations*

The arbitrage trader will be tested in homogeneous markets with $9$ GDZ traders and $9$ ZIP traders. The reason we chose for a market with ZIP traders is that they all individually have different price expectations. If all traders had the exact same price expectation arbitrage would not be possible.

Each test for the different market configurations will consist of $1000$ runs of $5$ periods which each last for $T = 200$ time steps. These parameters were chosen for the following reason. We chose a $1000$ runs so that we know that the algorithm consistently generates arbitrage profits and that we have a large enough sample size to see if the difference in profit is significant for different version of the algorithm. There are $5$ periods because preliminary testing has shown that behaviour stabilizes after around $3 - 4$ periods. Each period has $T = 200$ steps since this is around the average amount of time steps a ZIP algorithm stops trading due to the fact that there a no more feasible orders possible.

The beginning of a run starts with all traders and the exchange being reset. At the end of each period the allocations of the traders are reset but they still remember their expected prices and trade history from the previous period. Each time step every trader is allowed to act exactly once except if it posts a strategic order then it is allowed to post the first arbitrage order immediately after. This is achieved by selecting traders at random to act and without replacement every time step. The results of each run are averaged.

## 4. RESULTS

In tables and figures good $2$ is referred to as good $X$ and good $3$ is referred to as good $Y$. We differentiate between an completed trade an a failed trade in the tables. A completed arbitrage trade is a trade where both the strategic and the arbitrage orders resulted in a trade, albeit not for a profit. A failed arbitrage trade is where a trader has completed the arbitrage trade but was not able to buy/sell the good back before the end of the period.

The model was tested for $\theta_{min} = 0.1, 0.2, \cdots, 0.9$. Table I shows a comparison between the most important values of $\theta_{min}$. For $\theta_{min} < 0.4$ the lower the value of $\theta_{min}$ the more it engaged

TABLE I

ZIP-GDZ: RUITER BELIEF FUNCTION, COMPARISON BETWEEN DIFFERENT VALUES OF $\theta_{min}$.

$\theta_{min} = 0.4$

| | n | Profit | Target profit | Wait time |
|---|---|---|---|---|
| Total | 215 | -0.345 | 3.897 | 3.743 |
| Profit | 91 | 2.258 | 3.13 | 2.278 |
| Loss | 124 | -2.251 | 4.46 | 4.816 |
| Failed | 20.9 | - | 6.259 | 104* |

$\theta_{min} = 0.5$

| | n | Profit | Target profit | Wait time |
|---|---|---|---|---|
| Total | 87 | 0.512 | 1.497 | 1.847 |
| Profit | 64 | 1.352 | 1.434 | 0.921 |
| Loss | 23 | -1.806 | 1.668 | 4.403 |
| Failed | 4.3 | - | 1.873 | 115* |

$\theta_{min} = 0.6$

| | n | Profit | Target profit | Wait time |
|---|---|---|---|---|
| Total | 30 | 0.907 | 1.285 | 1.201 |
| Profit | 27 | 1.224 | 1.248 | 0.821 |
| Loss | 3 | -2.213 | 1.65 | 4.943 |
| Failed | 0.7 | - | 1.813 | 143* |

$\theta_{min} = 0.7$

| | n | Profit | Target profit | Wait time |
|---|---|---|---|---|
| Total | 11 | 0.99 | 1.198 | 1.174 |
| Profit | 9 | 1.169 | 1.175 | 0.928 |
| Loss | 2 | -2.648 | 1.653 | 6.205 |
| Failed | 0.2 | - | 1.762 | 155* |

Total is the average amount of completed arbitrage trades. Total is split into two rows profit and loss. Loss contains all trades that did not make a profit including the trades that broke even. We include break even trades to the loss category since the trader did not gain anything from the trade but still lost time. The profit column gives the average profit per arbitrage trade. The target profit indicates the average profit the trader initially wanted to gain when calculating $p_{opt}$. The wait time is the average amount of time steps in between the strategic trade and the arbitrage trade. The failed wait time is not the average wait time but the average time step the strategic order was placed.

in arbitrage but it also made increasingly less profit. For $\theta_{min} > 0.7$ profits no longer increased with higher values for $\theta_{min}$ but the amount of arbitrage trades dropped below $0.33\%$.

Table I shows that for larger values of $\theta_{min}$ the total amount of arbitrage trades decreases while average profit increases. It also shows an inverse relation between total target profit and actual total profit. An arbitrage trader is allowed to be wrong for a few trades as long as it makes more profit in the long run. Therefore we look at average aggregate profit per run to compare the performance for the different values of $\theta_{min}$. The average aggregate profit per

TABLE II

ZIP-GDZ: RUITER/GD BELIEF FUNCTION

| | GD belief function (1) | | | | Ruiter belief function (2) | | | |
|---|---|---|---|---|---|---|---|---|
| | n | Profit | Target profit | Wait time | n | Profit | Target profit | Wait time |
| Total | 258 | -0.414 | 4.915 | 4.051 | 87 | 0.512 | 1.497 | 1.847 |
| Profit | 96 | 3.312 | 4.805 | 2.9 | 64 | 1.352 | 1.434 | 0.921 |
| Loss | 162 | -2.62 | 4.98 | 4.732 | 23 | -1.806 | 1.668 | 4.403 |
| Failed | 26.5 | - | 6.881 | 98* | 4.3 | - | 1.873 | 115* |

Comparison between a GDZ trader that uses the GD belief function and a trader that uses the Ruiter belief function to calculate the probability of the arbitrage trade being accepted. Both versions had the same level of risk aversion at $\theta_{min} = 0.5$ and operated in the same market environment. Traders that use the GD belief function engage in a lot more arbitrage trades but are not profitable on average.

run is $-74.18, 44.54, 27.21$ and $10.89$ for the values $\theta_{min} = 0.4, 0.5, 0.6$ and $0.7$ respectively. $\theta_{min} = 0.5$ has the highest aggregate profit and will be used for future tables.

Table II shows that the GD belief function is a lot more optimistic and sets higher target profits. It engages in a lot more arbitrage traders and has a higher average profit. But it still has a negative total profit due to the higher percentage of losing trades ($62.8\%$ vs. $26.4\%$) and the fact that losses are bigger on average.

Table III shows the average wait time for profitable arbitrage trades is below $1$ indicating that most successful arbitrage trades are completed within the same time step. Naturally this is not possible for the trader that has to wait until the next time step. We would expect that the wait time for profitable trades would differ by $1$ since it misses the single time step advantage but surprisingly this is $2.6$ steps higher.

Did GDZ perform better than eGD? To test this we compare the excess goods at the end of the periods and the average utility. Table IV describes the average excess goods at the end of a period. We would expect GDZ to have a higher excess in money due to the profit it makes on arbitrage. Although GDZ shows slightly more excess per trader type and good the difference is not large enough to be significant. Figure 3 shows that the average utility levels are practically identical in shape and numbers. GDZ seems to have slightly higher average utility but the

TABLE III

ZIP-GDZ: DOUBLE MOVE

| | No double move | | | | Double move | | | |
|---|---|---|---|---|---|---|---|---|
| | n | Profit | Target profit | Wait time | n | Profit | Target profit | Wait time |
| Total | 69 | -0.307 | 1.603 | 4.045 | 87 | 0.512 | 1.497 | 1.847 |
| Profit | 31 | 1.539 | 1.612 | 3.559 | 64 | 1.352 | 1.434 | 0.921 |
| Loss | 38 | -1.789 | 1.596 | 4.436 | 23 | -1.806 | 1.668 | 4.403 |
| Failed | 10.8 | - | 1.564 | 87* | 4.3 | - | 1.873 | 115* |

Comparison between a GDZ trader that could post its arbitrage order immediately after posting its arbitrage order (double move) and GDZ trader that had to wait until the next time step. Note that this advantage was only granted when executing an strategic and arbitrage order back to back. Both versions had the same level of risk aversion at $\theta_{min} = 0.5$ and operated in the same market environment. Target profits were roughly the same but the trader that had no double move had a significantly higher percentage of lost trades ($55\%$ vs. $28\%$) and more failed trades.

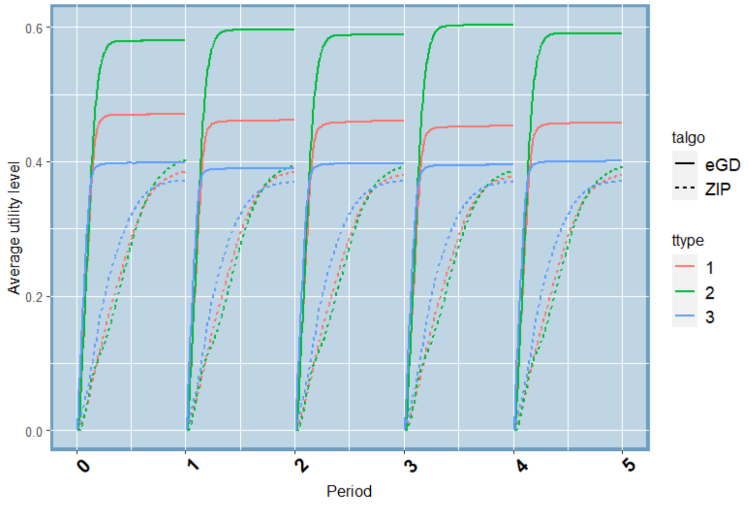
TABLE IV

NORMALIZED AVERAGE EXCESS

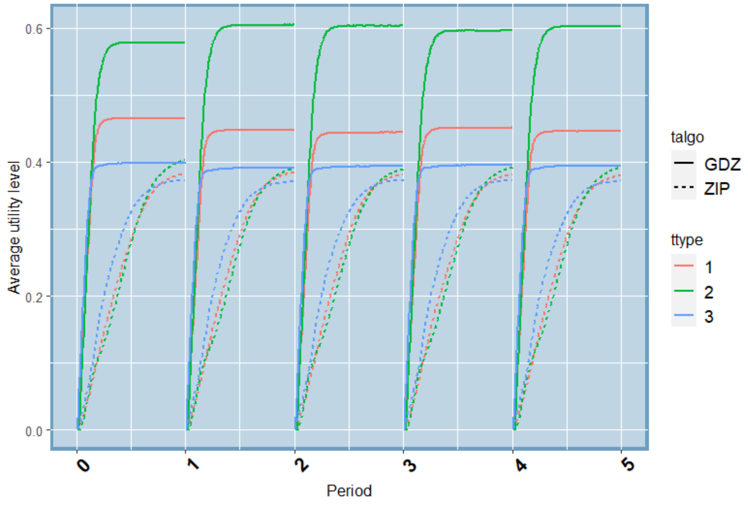
ZIP-GDZ VS. ZIP-EGD: RUITER BELIEF FUNCTION, $\theta_{min} = 0.5$

| Session: ZIP-GDZ | | | | | Session: ZIP-eGD | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Type | Algorithm | Money | X | Y | Type | Algorithm | Money | X | Y |
| 1 | GDZ | 1.469% | 0.53% | 1.83% | 1 | eGD | 1.29% | 0.11% | 1.71% |
| 1 | ZIP | 2.051% | 14.39% | 1.27% | 1 | ZIP | 2.12% | 15.42% | 1.23% |
| 2 | GDZ | 2.086% | 2.62% | 0.73% | 2 | eGD | 1.70% | 2.60% | 0.49% |
| 2 | ZIP | 4.461% | 1.02% | 32.74% | 2 | ZIP | 4.60% | 1.00% | 32.34% |
| 3 | GDZ | 3.254% | 1.75% | 0.83% | 3 | eGD | 3.06% | 1.63% | 0.70% |
| 3 | ZIP | 4.319% | 3.86% | 2.53% | 3 | ZIP | 4.49% | 3.98% | 2.55% |

Comparison between a GDZ trader and the algorithm it is based of, eGD. The table shows the normalized excess of the goods at the end of the period. Excess is defined as the amount of goods a trader could "give away" at the end of the period without using utility. The less excess an algorithm has the more efficient it is.




(a) ZIP-eGD




(b) ZIP-GDZ: Ruiter belief, $\theta_{min} = 0.5$

FIGURE 3.—Average utility level per period. GDZ and eGD traders consistently outperform ZIP traders and reach optimal utility levels much faster. Interestingly, type $II$ traders have a much better performance than types $I$ and $III$ for eGD and GDZ traders. Theoretically they should get roughly the same utility. The reason for this is that they start with good $Y$. At the start of the period prices are higher than the expected equilibrium $(40, 20)$ and not in the $2:1$ ratio. This means that good $Y$ is overvalued at the beginning of the period allowing a trader of type $II$ to gain an unfair amount of money compared to the traders of other types. The reason ZIP traders do not have this is that they select actions at random so by the time they have sold all their goods prices are already back to normal.

difference is not significant. Hence the differences between eGD and GDZ are not large enough to say that one performs better than the other.

## 5. DISCUSSION

To avoid repeating the counter example (bid/ask, higher/lower) all explanations below are written from the perspective of an arbitrage trader buying a good with the intent of selling for a higher price.

### 5.1. *Risk Aversion*

The level of risk aversion of a GDZ trader is given by $\theta_{min}$. The higher the value for $\theta_{min}$ the less risk the trader is willing to take. There is a clear relation between risk aversion and target profit. The less risk the trader is willing to take the less it engages in arbitrage opportunities and sets lower target profits. While this does lead to a higher percentage of profitable trades the number of arbitrage trades declines rapidly. We believe that an successful arbitrage trader should take sufficient risk such that there are enough opportunities but not so much that it consistently makes losses. Also the number of failed trades is important. Making a small loss does not affect utility levels if you have an excess of money. But having a failed trade at the end of the period means the trader was not able to gain the lost utility back form the strategic trade. Therefore a markets with high risk arbitrage traders are less efficient due to the increased excess. The value for $\theta_{min}$ affects the number of arbitrage trades but also the profitability of it. Therefore it should be calibrated in such a way that it generates the most aggregate profit. If the trader gains enough profit through arbitrage trades it has the extra capital to spend on utility increasing goods.

### 5.2. *Belief function*

Table II shows a significant difference when using a different belief function. The GD belief function (1) performs well when it is used to estimate probabilities that are close to observed prices and is therefore good as a means to calculate the equilibrium price. However it overestimates the probability of bids or asks being accepted at the extremes. This led to the algorithm buying or selling goods for the strategic order that were far from equilibrium/observed prices. Then the arbitrage order was quickly rejected, often with a price lower than the original price, causing the arbitrage trader to consistently make losing trades.

The Ruiter belief function (2) takes the possibility of rejection into account by calculating the probability the bid will be accepted before it is rejected. Changing the belief function to the one Ruiter proposed caused the algorithm to take strategic orders that were closer to observed prices and have more realistic target profits. Although this decreased the number of arbitrage trades it now consistently makes a profit. These findings show the importance of a realistic belief function.

### 5.3. *Speed*

Table III shows that a GDZ trader, which is allowed to post its arbitrage trade immediately after its strategic trade, outperforms the GDZ trader that has to wait. We believe this advantage benefits the trader in two ways. Say the GDZ trader has bought good X for 35 to sell it back for 40. First advantage is that it has more opportunities to act. Second advantage is that it can move twice in a row. This means that the double move trader has a guaranteed chance its first arbitrage order will be in the order book meaning that it automatically has one less chance for rejection. This is because a non double move trader can be rejected before it even posts its first arbitrage order if at the next time step another trader has already posted an ask for 38. This advantage is clearly reflected by the fact that the average profitable trade wait time is less than 1 meaning most profitable arbitrage trades are completed within the same time step. We see that not having the double move does not impact behaviour as the total trades and target profit are about the same but the waiting time significantly increased for successful trades leading to more rejection and affecting profitability. From these results we can conclude that speed is an important factor for the profitability of an arbitrage trader. This agrees with the findings of Hanifan et al. (2021) which show that speed is an important factor for the performance of a trading agent.

### 5.4. *Future improvements*

The findings of Hanifan et al. (2021) have shown that urgency can have a large impact on the performance of an trader. This is especially the case with arbitrage traders. From the results we see that optimistic traders have a high number of rejected arbitrage trades. If they can not sell their arbitrage good back before the end of the period they lose utility. Therefore as the period ends, arbitrage traders should adjust their profit margins more aggressively, even taking large losses if it means they can regain that utility. We could model this sense of urgency by incorporating a process similar to the pace parameter of Hanifam's ZIPP algorithm. This pace parameter decreases the profit margin the longer it has to wait until the order is accepted or the closer it gets to the end of the period.

Another solution for rejected trades is to make the algorithm engage in less arbitrage opportunities the closer they are to the end of the period. We can model this in two ways. Firstly, Trading volume declines the closer you get to the end of the period. If there is less volume there is a lower chance your order will be accepted. We could model urgency by measuring the quantities of orders of the last $n$ time steps as volume $V^j$ for each good $j$. Then estimate maximum volume $V^j_{max}$ from a previous period. Use $\frac{V^j}{V^j_{max}}$ as a weight for the belief function. This will reduce the amount of times the trader will engage in arbitrage when there is less volume.

Secondly, Right now GDZ has a $20\%$ to look at an arbitrage opportunity. We can make this percentage a function of $u = \frac{T-t}{T}$ where $t$ is the current time step and $T$ is the final time step. In this way we can scale this percentage down using $f(u) = P[\text{arbitrage}]$ as the period comes to an end. By trying different forms of the function $f(u)$ we hope to decrease the number of rejected trades.

We except these changes to have a significant effect on the profitability of an arbitrage trader. It would be interesting to see if incorporating these changes would make the algorithm profitable enough to gain extra utility from the strategy.

## 6. CONCLUSION

This paper has shown that an arbitrage strategy has the potential to be a viable strategy for a trading agent in the Scarf economy. The current implementation does not generate enough profits to consistently gain extra utility from the strategy. The most important factors for profitability are: a realistic belief function, conservative profit margins and speed. From the results we can see that setting higher target profits gives longer wait times which increases the percentage of losing trades. Also more optimistic traders got stuck with commodities that did not yield utility. Although we were not able to model the results show evidence that the feasibility of arbitrages depend critically on time.

During the developing of the CDA and implementing the algorithms we faced a lot of design choices that could have a very large impact on the end result. This makes it difficult to compare the findings of this paper to previous research as the same choices the other researchers had to make are not detailed in the literature. We conclude that testing trading agents in a self developed environment is useful for developing ideas about profitable strategies but that it is not possible to generalize them due to the number of specific design choices.

## REFERENCES


ANDERSON, CHRISTOPHER M., CHARLES R. PLOTT, KEN ICHI SHIMOMURA, AND SANDER GRANAT (2004): "Global instability in experimental general equilibrium: The Scarf example," *Journal of Economic Theory*, 115, 209–249. [1, 2, 3, 4, 5, 8]

CLIFF, DAVE AND JANET BRUTEN (1997): "Minimal-Intelligence Agents for Bargaining Behaviors in Market-Based Environments," Tech. rep. [2, 6]

GJERSTAD, STEVEN AND JOHN DICKHAUT (1998): "Price Formation in Double Auctions," *Games and Economic Behavior*, 22, 1–29. [3, 6]

GODE, DHANANJAY K AND SHYAM SUNDER (1993): "Allocative Efficiency of Markets with Zero-Intelligence Traders: Market as a Partial Substitute for Individual Rationality," Tech. Rep. 1. [2]

HANIFAN, HENRY, BEN WATSON, JOHN CARTLIDGE, AND DAVE CLIFF (2021): "Time Matters: Exploring the Effects of Urgency and Reaction Speed in Automated Traders," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Springer Science and Business Media Deutschland GmbH, vol. 12613 LNAI, 149–170. [3, 17]

RUITER, A.G.J.M. (2018): "Price discovery with fallible choice," . [4, 6]

SCARF, HERBERT (1960): "Some Examples of Global Instability of the Competitive Equilibrium," Tech. Rep. 3. [3, 4]

SMITH, VERNON L (1962): "An Experimental Study of Competitive Market Behavior," *The Journal of political economy*, 70, 111–137. [2]

TESAURO, GERALD AND JONATHAN L BREDIN (2002): "Strategic Sequential Bidding in Auctions using Dynamic Programming," Tech. rep. [3]

VYTELINGUM, PERUKRISHNEN (2006): "The Structure and Behaviour of the Continuous Double Auction," Tech. rep. [3]

# APPENDIX A:
## CODE

The code for the simulated CDA and the different trading algorithms can be found on GitHub at https://github.com/thomas-vdz/BScThesisCDA